A Shifted Block Lanczos Algorithm for Solving Sparse Symmetric Generalized Eigenproblems

Roger G. Grimes[1], John G. Lewis[1], and Horst D. Simon[2]

Report RNR-91-012, July 1991

NAS Systems Division
Applied Research Branch
NASA Ames Research Center, Mail Stop T045-1
Moffett Field, CA 94035

July 18, 1991

Abstract. We describe an "industrial strength" algorithm for solving sparse symmetric generalized eigenproblems. The algorithm has its foundations in known techniques in solving sparse symmetric eigenproblems, notably the spectral transformation of Ericsson and Ruhe and the block Lanczos algorithm. However, the combination of these two techniques is not trivial; there are many pitfalls awaiting the unwary implementor. The focus of this paper is identifying those pitfalls and avoiding them, leading to a "bomb-proof" algorithm that can live as a black box eigensolver inside a large applications code. The code that results comprises a robust shift selection strategy and a block Lanczos algorithm that is a novel combination of new techniques and extensions of old techniques.

[1]Scientific Computing and Analysis Division, Boeing Computer Services, M/S 7L-21, Seattle, WA 98124

[2]The author is an employee of Computer Sciences Corporation. This work has been supported through NASA Contract NAS 2-12961.

A SHIFTED BLOCK LANCZOS ALGORITHM FOR SOLVING SPARSE SYMMETRIC GENERALIZED EIGENPROBLEMS

ROGER G. GRIMES*, JOHN G. LEWIS* AND HORST D. SIMON†

Abstract. We describe an "industrial strength" algorithm for solving sparse symmetric generalized eigenproblems. The algorithm has its foundations in known techniques in solving sparse symmetric eigenproblems, notably the spectral transformation of Ericsson and Ruhe and the block Lanczos algorithm. However, the combination of these two techniques is not trivial; there are many pitfalls awaiting the unwary implementor. The focus of this paper is identifying those pitfalls and avoiding them, leading to a "bomb-proof" algorithm that can live as a black box eigensolver inside a large applications code. The code that results comprises a robust shift selection strategy and a block Lanczos algorithm that is a novel combination of new techniques and extensions of old techniques.

Key Words. Lanczos algorithm, sparse eigenvalue problems, structural analysis, symmetric generalized eigenvalue problem, orthogonalization methods

AMS(MOS) subject classification. 65F15, 15A18, 65F50, 73K99

Contents.

* Scientific Computing and Analysis Division, Boeing Computer Services, M/S 7L-21, Seattle, WA 98124

† Numerical Aerodynamic Simulation (NAS) Systems Division, NASA Ames Research Center, Mail Stop T-045-1, Moffett Field, CA 94035.The author is an employee of Computer Sciences Corporation. This work was funded in part through NASA Contract Number NAS 2-12961.

1. Introduction. The Lanczos algorithm [21] is widely appreciated in the numerical analysis community [6, 7, 8, 14, 15, 17, 22, 28, 29, 31, 34, 36] as a very powerful tool for extracting some of the extreme eigenvalues of a real symmetric matrix H, i.e. to find the largest and/or smallest eigenvalues and vectors of the symmetric eigenvalue problem

$$Hx = \lambda x.$$

However, the usefulness of the algorithm is often misunderstood. It is often believed, for example, that the algorithm can be used directly to find the eigenvalues at both ends of the spectrum (both largest and smallest in value). In fact, many applications result in eigenvalue distributions that only allow effectively extracting the eigenvalues at one end of the spectrum. For example, the smallest eigenvalues are usually the ones of interest in structural engineering vibration problems. Typical eigenvalue distributions in these problems have small eigenvalues of order unity with separations $|\lambda_{i+1} - \lambda_i|$ also of order unity, apparently well-separated. However, for physical reasons the largest eigenvalues of these problems are very large, say $\mathcal{O}(10^{10})$. The convergence rates for the eigenvalues is determined primarily by the relative separation $\frac{|\lambda_{i+1}-\lambda_i|}{|\lambda_n-\lambda_1|}$, which for the smallest eigenvalues is $\mathcal{O}(10^{-10})$. We expect to see and do find very slow convergence to the small eigenvalues. The fundamental characteristic that is usually ignored is the dependence of convergence on *relative* separation between eigenvalues.

When the eigenvalues at one end are much closer together than the eigenvalues at the other end, only the eigenvalues at the well-separated end converge at a reasonable rate.

It is also often believed that the Lanczos algorithm can be applied to the generalized symmetric problem

$$Hx = \lambda Mx$$

by using the naive reduction to standard form [16, 31]: factor M into its Cholesky decomposition $M = LL^T$ and then solve the ordinary eigenproblem $L^{-1}HL^{-T}y = \lambda y$. Suppose that we applied this algorithm to the *vibration* problem of structural engineering,

$$Kx = \lambda Mx, \tag{1}$$

where K is the stiffness matrix and M is the mass matrix. We would fail abysmally for three separate reasons:

- M is very often semi-definite — it may admit no Cholesky factorization.
- even when M can be factored, typically the eigenvalues that are desired are the smallest, but they are very badly separated.
- the eigenvectors x must be computed by a back transformation $x = L^{-T}y$. When it exists L is usually poorly conditioned, which can lead to considerable numerical error in the back transformation.

When K is positive definite, the vibration problem can be addressed by applying the usual reduction to the reciprocal problem:

$$Kx = \lambda Mx \Leftrightarrow Mx = \frac{1}{\lambda}Kx \Leftrightarrow L^{-1}ML^{-T}y = \mu y, \tag{2}$$

where L is the Cholesky factor of K and $\mu = \frac{1}{\lambda}$. Often this is sufficient as a cure for the first two problems in (1), because the reciprocals of the eigenvalues are well-separated. Eigenanalysis codes in structural engineering packages [23, 26] have been built upon this transformation. However, it still suffers from the third problem. More importantly, this transformation is still inadequate in the following circumstances:

- the model has rigid body modes — K is positive semi-definite and has no Cholesky decomposition.
- a considerable number of eigenvalues are desired.
- the eigenvalues wanted are not the smallest eigenvalues.

The second and third problems are essentially the same, in that the Lanczos algorithm normally does not compute good approximations to interior eigenvalues until it has computed *all* the eigenvalues between the interior eigenvalues and the nearest end of the spectrum.

Applications with these characteristics do arise. For example, structures with rigid body modes are common in aerospace applications — the rigid body modes correspond to the fact that the structure, say an airplane, is free to translate or rotate in space without affecting the vibration analysis. The stiffness matrix typically has a six-dimensional nullspace of rigid body modes. Detailed analyses of structures may require more than just a few eigenvalues and vectors. One of the test problems used here is an analysis of a nuclear reactor containment floor, where more than 200 eigenpairs were needed to adequately model

the response of the structure to a simulated earthquake. Another problem we have analyzed is a model of a large industrial ventilating fan mounted on a large concrete platform. The real physical structure resonated at the normal operating speed of the fan. Corrective measures depended on getting good approximations to the eigenvalues near the fan's rotational rate, eigenvalues that are in the interior of the spectrum. (Blindly applying standard engineering fixes to this particular problem would have created a newsworthy catastrophe.)

There is a more elaborate transformation of the problem, the *spectral transformation* of Ericsson and Ruhe [14], which is capable of treating all of these difficulties. The spectral transformation is discussed in detail in §2, where we discuss extension of the standard algorithm to buckling as well as vibration problems. The general idea behind the spectral transformation comes from considering the shifted problem $(K - \sigma M)x = (\lambda - \sigma)Mx$. If we invert $(K - \sigma M)$ we transform the eigenvalues nearest the *shift* σ into the largest and well-separated eigenvalues of the reciprocal problem. Normally we need only to choose a shift σ near the eigenvalues we want. When the number of eigenvalues is large, the reduced convergence rate of the eigenvalues farthest from σ makes it worthwhile to choose additional shifts (and factorizations) in order to search through the spectrum.

Formally the only key characteristic of the shift σ is that it is not allowed to be equal to an eigenvalue of the problem. In such a case the shifted operator is singular. It is probably impossible to determine numerically that the operator in singular. In fact, avoiding singularity is an issue for the choice of shifts, but it can be confined mostly to the choice of the very first shift. The choice of the initial shift is an interesting problem; it is discussed in §3.4.

In general, a well-chosen shift allows us to compute tens of eigenvalues with a single Lanczos run. A poorly chosen shift may only be useful for computing a single cluster of eigenvalues or perhaps no interesting eigenvalues at all. In addition, there is a complicated tradeoff between the cost of a Lanczos run, which increases nonlinearly with increasing numbers of steps, and the cost of computing a new shift and its concomitant factorization. As an example, we consider the oceanography model (matrix PLAT1919 in the Harwell/Boeing sparse matrix collection [11], with four different paradigms for choosing shifts:

- the heuristic described in this paper;
- a conservative modification of this heuristic;
- an aggressive modification of this heuristic;
- no shifting – attempting to compute all 200 eigenvalues with a single factorization.

The following table contains the salient results for these choices, demonstrating the virtues of carefully choosing shifts. The shift factor, normally two, is described in §3.2; it is the multiplier applied to k or to δ in obtaining the next shift. All of these analyses begin with a Lanczos run applied to the matrix $A - .0001I$.
(These results were obtained on a Sun 4/490 workstation. The code used a blocksize of five. Execution cost is the sum of cpu and i/o processor seconds.)

Shifting can provide reliability as well as efficiency. Each factorization provides eigenvalue location information in the form of *matrix inertias* (see §3.1). The collected inertias from a series of well-chosen shifts can provide an independent guarantee on the success of the eigenvalue computation. Alternately, the inertia information can be used to drive the

TABLE 1
Computing the 200 Lowest Eigenvalues in [.0001, .24] *of PLAT1919*

choice of shift	shift factor	number of Lanczos runs	total number of Lanczos steps	execution cost
normal	2.0	8	145	317
conservative	1.0	13	187	418
aggressive	3.0	6	131	323
no shifting		1	318	6637

choice of further shifts and Lanczos runs to ensure that all of the desired eigenvalues have been computed. A well-designed strategy for choosing shifts can lead to an algorithm that is both robust and efficient. Our heuristic strategy for choosing shifts is discussed in §3.

Our goal is a code that can serve as a "black-box" eigenextraction routine in large applications codes. As such, there are at least two additional problems that must be faced at the outset. First, eigenvalues cannot be assumed to be simple. Multiple eigenvalues are common in a number of applications. We use in our test suite three problems, all captured from real applications, which have multiple eigenvalues in configurations ranging from a few doubletons and triples to all eigenvalues appearing in pairs to several very large clusters of equal eigenvalues. Rigid body modes usually occur as a cluster of six zero eigenvalues. The code must be prepared for such situations. Our approach is three-fold. First, our shifting strategy is prepared to continue looking at a small piece of the spectrum until it has determined the full multiplicity of the eigenvalues therein. Second, the shifting scheme and the Lanczos algorithm interact to ensure that we find an orthogonal basis for the invariant subspace for each cluster (see §4.3.3). Most importantly, we use a *block* version of the Lanczos algorithm. Provided that we have been able to choose a block size as large as the largest multiplicity of any cluster we will encounter, we expect that the Lanczos algorithm will compute the full multiplicities of each cluster without any intervention from the shifting strategy.

The block Lanczos algorithm also handily addresses another difficulty posed by its use inside applications codes. Applications codes often use general representations for their data, even when particular machine architectures would allow or favor alternatives. Even with the increases in main memory over the last decade it is common for general applications codes to represent their matrices as "out-of-core". As such there is a considerable cost in accessing a matrix. The block Lanczos code substitutes, almost on a one for one basis, matrix-block multiplies and block solves for matrix-vector products and simple solves. The effect is to decrease the input/output cost essentially by the block size.

Our production eigenextraction code is a synthesis of the ideas of the spectral transformation and the block Lanczos algorithm. Following this introduction we present the details of the spectral transformation and of the block Lanczos algorithm. In §2 we begin to address the effects of the generalized problem on the recurrence. We explain how to form the shifted and inverted operators for both vibration and buckling analysis, and what modifications to the Lanczos recurrence result. With the exception of the development of a spectral trans-

formation for buckling problems, our presentation is quite standard and is provided for the reader not already familiar with these results.

With these basic tools in hand, we present our heuristic shifting strategy in §3. There are eight subsections: a discussion of *trust intervals* and matrix inertias, our basic tools for robustness; our heuristic for choosing a shift in a generic case; the idea of *sentinels*, a tool for ensuring orthogonality of invariant subspaces; our heuristics for choosing an initial shift; our heuristics for determining how to expand the primary trust interval; our treatment of various special and pathological cases; and last, the modifications needed for the buckling problem.

The special characteristics of our block Lanczos algorithm are discussed in §4. This considers the effects due to the spectral transformation. One major problem is that vectors must be orthonormalized with respect to an inner product defined by a positive definite matrix M. Although M-orthogonality represents little difficulty in theory, the setting of these problems requires more thought. A discussion of the issues associated with implementing M-orthonormalization of vectors in the basic block Lanczos algorithm is given in §4.1, as are the further precautions needed to allow cases where M induces only a semi-norm.

The block Lanczos recurrence by itself produces only a block tridiagonal matrix T. In §4.2 we describe how to compute eigenvalue and vector approximations, and error bounds on these approximations, from T and the Lanczos vectors. §4.3 contains our approach for dealing with the effects of finite precision arithmetic, and particularly for addressing the loss of orthogonality in the Lanczos vectors. Here we present a novel combination of various reorthogonalization schemes, which are combined to work effectively with the unusual distributions of eigenvalues that result from the spectral transformation. §4 concludes with discussions of when to end and how to start the recurrence. The integration of all of these techniques provides a block Lanczos recurrence that will effectively find a limited number of eigenvalues and corresponding eigenvectors of a spectrally transformed operator.

We close with a summary of the environment used in our numerical experiments and discussions of the overall behavior of our code on a small set of eigenproblems obtained from applications codes.

2. The Spectral Transformation Block Lanczos Algorithm. The eigenvalue problem in vibration analysis is given as

$$Kx = \lambda Mx, \tag{3}$$

where K and M are symmetric matrices, and M is positive semidefinite. Usually only the smallest eigenvalues of (3) are wanted. Although these lie at one end of the spectrum, they are commonly very poorly separated. In a typical distribution of eigenvalues in a vibration problem, the interesting eigenvalues have very poor relative separation, rarely better than $\mathcal{O}(10^{-6})$. At the same time, the largest eigenvalues, which are uninteresting, have very good separation. *A priori* estimates for the rate of convergence predict very slow convergence at the desired end of the spectrum and fast convergence at the other end, both of which are observed in practice. We can obtain rapid convergence to the desired eigenvalues by using the *spectral transformation* [14, 26] of (3).

2.1. The Spectral Transformation for Vibration Problems. Consider the problem

$$M(K-\sigma M)^{-1}Mx = \mu Mx, \tag{4}$$

where σ, the shift, is a real parameter. Assume for the moment that M is positive definite; the complications introduced by a semidefinite M will be discussed later. It is easy to verify that (λ, x) is an eigenpair of (3) if and only if $(\frac{1}{\lambda-\sigma}, x)$ is an eigenpair of (4). Hence, the transformation of the eigenvalue problem from (3) to (4) does not change the eigenvectors, and the eigenvalues are related by

$$\mu = \frac{1}{\lambda-\sigma}. \tag{5}$$

The form of the spectral transformation, especially when compared to (2), may seem complicated. The form is dictated by our need to be able to apply the Lanczos algorithm even in cases when M is semidefinite. Other advantages of this form are well-documented in [37].

The main advantage of applying the Lanczos algorithm to (4) instead of (3) becomes clear when the effect of the spectral transformation on the spectrum is considered. Figure 1 gives the general shape of the transformation. The results in Table 2 demonstrate the effect of the transformation in detail. These are the values obtained using the initial shift described in §3.4; the generalized eigenproblem is the model of a nuclear reactor containment floor, given by the stiffness and mass matrices BCSSTK26 and BCSSTM26, respectively, from the Harwell-Boeing sparse matrix collection[11]. (We denote the generalized eigenproblem by BCSST_26.)

Relative separation is affected dramatically. The smallest eigenvalues are transformed from eigenvalues with extremely poor relative separation to eigenvalues with good relative separation, even though their absolute separation is decreased. More important in this transformation is that the eigenvalues far from the shift are transformed to poorly separated values near zero. This spread of the eigenvalues ensures rapid convergence to the eigenvalues near σ. This is illustrated in Figure 1, where the desired eigenvalues are well off-scale, in the region that appears to be suffering compression, and yet the relative separation is enormously improved. This example clearly demonstrates that the shift does not have to be very close in an absolute sense to work well.

The primary price for this rapid convergence is the cost of a factorization of $K-\sigma M$. Of course, we never form the actual transformation $M(K-\sigma M)^{-1}M$ explicitly — it is almost certainly a dense matrix. Instead the transformation is realized implicitly as a sequence of operations, in which we compute MQ for a block of vectors Q, or solve the linear systems $(K-\sigma M)X = Q$. In practice, these operations are realized by separate subroutines. This modularity allows tuning the matrix factorization and multiplication routines to the class of problem under consideration.

An additional complication introduced by the spectral transformation is that we still are faced with a generalized symmetric eigenproblem; we must generalize the Lanczos algorithm itself. We will make this generalization in three steps. We will first consider the ordinary block Lanczos algorithm for a symmetric matrix H, during which we will establish our

FIG. 1. *Vibration Spectral Transformation*

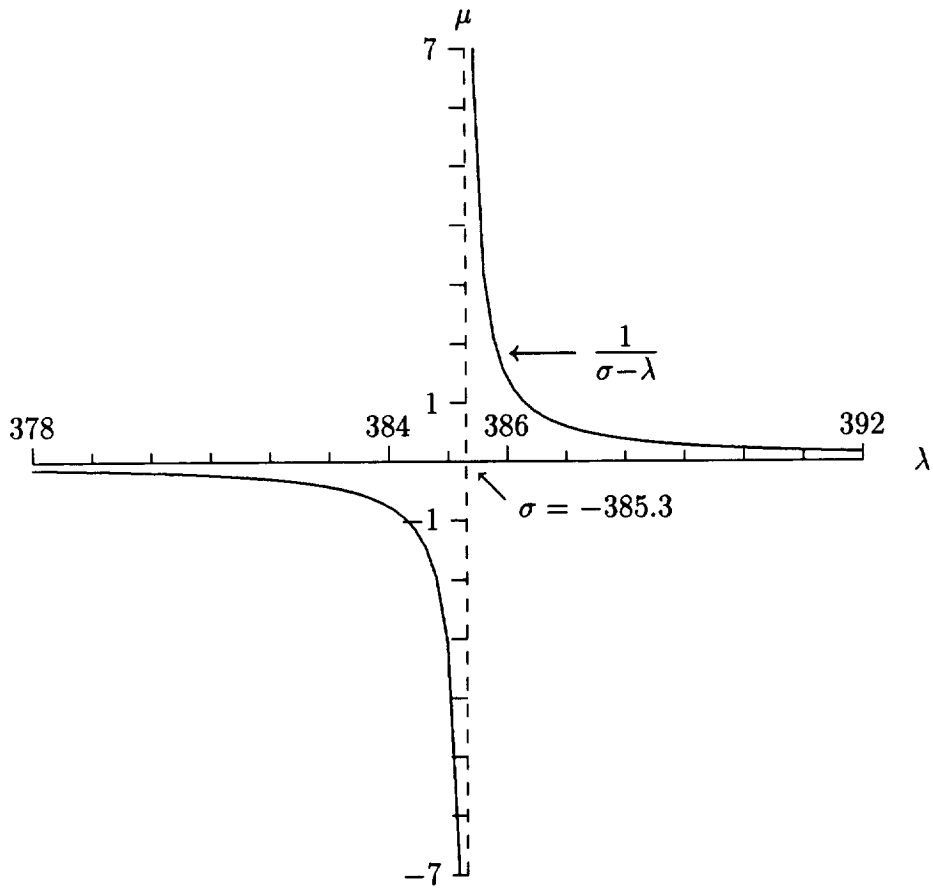


TABLE 2
Vibration Spectral Transformation of BCSST_26, $\sigma_1 = 385.3$

			original		transformed	
i	$\lambda(i)$	$\mu(i)$	gap	relative gap	gap	relative gap
1	4.6×10^{3}	2.4×10^{-4}	6.4×10^{3}	1.2×10^{-11}	1.4×10^{-4}	6.0×10^{-1}
2	1.1×10^{4}	9.4×10^{-5}	2.5×10^{2}	4.6×10^{-13}	2.2×10^{-6}	9.2×10^{-3}
3	1.1×10^{4}	9.2×10^{-5}	2.5×10^{2}	4.6×10^{-13}	2.2×10^{-6}	9.2×10^{-3}
4	1.5×10^{4}	7.0×10^{-5}	3.4×10^{3}	6.3×10^{-12}	2.1×10^{-5}	8.7×10^{-2}
⋮	⋮	⋮	⋮	⋮	⋮	⋮
1919	2.4×10^{14}	4.2×10^{-15}	2.9×10^{13}	5.4×10^{-2}	5.9×10^{-16}	2.5×10^{-12}
1920	3.0×10^{14}	3.3×10^{-15}	3.6×10^{11}	6.7×10^{-4}	3.9×10^{-18}	1.7×10^{-14}
1921	3.1×10^{14}	3.3×10^{-15}	3.6×10^{11}	6.7×10^{-4}	3.9×10^{-18}	1.7×10^{-14}
1922	5.4×10^{14}	1.8×10^{-15}	2.4×10^{14}	4.4×10^{-1}	1.4×10^{-15}	6.0×10^{-12}

notation. Next we consider a direct generalization of the Lanczos algorithm for an arbitrary generalized symmetric eigenproblem $Hx = \lambda Mx$, where we assume temporarily that M is positive definite. In these first two steps the issue of shifting disappears for the moment. In a third step we will move from this general form of the algorithm to the much more effective form that is possible when H has the special form used in the spectral transformation.

2.2. Basic Block Lanczos Algorithm. Consider first the ordinary eigenvalue problem

$$Hx = \lambda x,$$

where H is a real symmetric linear operator. An important characteristic of the Lanczos algorithm is that H is not required explicitly. All that is required is a subroutine that computes Hy for a given vector y. The block Lanczos iteration with *blocksize* p for an $n \times n$ matrix H is given in Figure 2.

FIG. 2. *Basic Block Lanczos Algorithm*

Initialization:

Set $Q_0 = 0$
Set $B_1 = 0$
Choose R_1 and orthonormalize the columns of R_1 to obtain Q_1

Lanczos Loop:

For $j = 1, 2, 3 \ldots$ do
  Set $U_j \quad = HQ_j - Q_{j-1}B_j^T$
  Set $A_j \quad = Q_j^T U_j$
  Set $R_{j+1} = U_j - Q_j A_j$
  Compute the orthogonal factorization $Q_{j+1}B_{j+1} = R_{j+1}$
End loop

The matrices Q_j, U_j, R_j for $j = 1, 2, \ldots$ are $n \times p$, while A_j and B_j are $p \times p$, with A_j symmetric. The matrices A_j and B_j are the generalizations of the scalars α_j and β_j in the ordinary Lanczos recurrence; R_{j+1} is the *residual* at the $j+1$-st step. The matrices Q_{j+1} and B_{j+1} are defined in the last step of the recurrence by the orthogonal (QR) factorization of R_{j+1}, so that B_{j+1} is upper triangular, and the columns of Q_{j+1} are orthonormal.

This formulation of the Lanczos loop is the one least susceptible to round off errors [30] and is the form that should be used in computation. In exact arithmetic, however, U_j and R_{j+1} can be eliminated from the Lanczos loop and the recurrence formula becomes

$$Q_{j+1}B_{j+1} = HQ_j - Q_j A_j - Q_{j-1}B_j^T. \tag{6}$$

This three-term recurrence is what appears in earlier references to the block Lanczos algorithm. The differences in formulation do not change the theoretical properties of the

recurrence. In particular, it is shown in [6, 17] that the combined column vectors of the matrices $Q_1, Q_2, \ldots Q_j$, the so called *Lanczos vectors*, form an orthonormal set. The computational efficiency of the Lanczos algorithm rests on the fact that these vectors can be computed with a simple recurrence and with a fixed amount of work per iteration step.

The blocks of Lanczos vectors collectively form an $n \times jp$ matrix $\mathcal{Q}_j$, where

$$\mathcal{Q}_j = [Q_1, Q_2, Q_3, \ldots Q_j].$$

The algorithm also defines a $jp \times jp$ block tridiagonal matrix T_j:

$$T_j = \begin{pmatrix} A_1 & B_2^T & 0 & \ldots & 0 \\ B_2 & A_2 & B_3^T & \ldots & 0 \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ldots & B_{j-1} & A_{j-1} & B_j^T \\ 0 & \ldots & 0 & B_j & A_j \end{pmatrix}.$$

Since the matrices B_j are upper triangular, T_j is a band matrix with half band width $p+1$ (rather than $2p$, if the B_j were full). The first j instances of formula (6) can be combined into a single formula:

$$H\mathcal{Q}_j = \mathcal{Q}_j T_j + Q_{j+1} B_{j+1} E_j^T. \tag{7}$$

Here E_j is an $n \times p$ matrix whose last $p \times p$ block is the $p \times p$ identity matrix and which is zero otherwise. Formula (7) is a compact way of expressing the Lanczos recurrence, and will be used throughout this discussion.

By premultiplying (7) by $\mathcal{Q}_j^T$ and using the orthogonality of the Lanczos vectors, it follows that

$$\mathcal{Q}_j^T H \mathcal{Q}_j = T_j.$$

Hence T_j is the orthogonal projection of H onto the subspace spanned by the columns of $\mathcal{Q}_j$. For $p = 1$, this space is called the *Krylov subspace* $K_j(H; q_1)$. It can be shown by induction that

$$span(\mathcal{Q}_j) = span(Q_1, HQ_1, H^2Q_1, \ldots H^{j-1}Q_1),$$

where *span (.)* denotes the subspace spanned by the columns of the matrices involved. From a different perspective, the (block) Lanczos algorithm is a method for constructing an orthonormal basis for the (block) Krylov subspace determined by H and Q_1. This basis of Lanczos vectors is distinguished by the fact that the orthogonal projection of H onto the (block) Krylov subspace is given by a (block) tridiagonal matrix. Hence the eigenvalues of T_j are the Rayleigh-Ritz approximations from $span(\mathcal{Q}_j)$ to the eigenvalues of H. In addition, if s is an eigenvector of T_j, the vector $y = \mathcal{Q}_j s$ is an approximate eigenvector of H. Viewed in this form, the Lanczos algorithm replaces a large and difficult eigenvalue problem involving H by a small and easy eigenvalue problem involving the block tridiagonal matrix T_j.

How good are the approximations obtained by solving the block tridiagonal eigenvalue problem involving the matrix T_j? An *a posteriori* bound on the residual can be obtained as follows: Let θ, s be an eigenpair for T_j, i.e.,

$$T_j s = s\theta,$$

and let

$$y = \mathcal{Q}_j s,$$

then

$$\begin{aligned} \|Hy - y\theta\|_2 &= \|H\mathcal{Q}_j s - \mathcal{Q}_j s\theta\|_2 \\ &= \|\mathcal{Q}_j T_j s + Q_{j+1}B_{j+1}E_j^T s - \mathcal{Q}_j s\theta\|_2 = \|Q_{j+1}B_{j+1}E_j^T s\|_2 \\ &= \|B_{j+1}E_j^T s\|_2 = \|B_{j+1}s_j\|_2, \end{aligned} \tag{8}$$

where s_j are the last p components of the eigenvector s. The quantity $\|B_{j+1}s_j\|_2$ can be computed without computing the approximate eigenvector y. Hence, with some modifications described in §4.2, (8) provides an inexpensive *a posteriori* error bound.

Formula (8), however, does not guarantee that good approximations to eigenpairs will appear quickly. Such *a priori* estimates are provided by the Kaniel-Paige-Saad theory. Parlett [31] gives the most detailed discussion for the single vector case ($p = 1$). The generalizations to the block case were originally derived by Underwood [17]. Extensions to both of these presentations can be found in [35].

2.3. The Spectral Transformation Block Lanczos Algorithm. The next step is to consider the generalized symmetric eigenproblem $Hx = \lambda Mx$. Were we to reduce the problem to standard form by factoring M, the three term recurrence (6) would become

$$Q_{j+1}B_{j+1} = M^{-1/2}HM^{-1/2}Q_j - Q_jA_j - Q_{j-1}B_j^T. \tag{9}$$

If we premultiply (9) by $M^{1/2}$ and make the transformation of variables $\hat{Q}_j = M^{-1/2}Q_j$, (9) becomes

$$\begin{aligned} M\hat{Q}_{j+1}B_{j+1} &= M^{1/2}M^{-1/2}H\hat{Q}_j - M\hat{Q}_jA_j - M\hat{Q}_{j-1}B_j^T \\ &= H\hat{Q}_j - M\hat{Q}_jA_j - M\hat{Q}_{j-1}B_j^T. \end{aligned} \tag{10}$$

Again the value of the transformation may not be clear. The matrices $\hat{Q}_j$ are now M-orthogonal, rather than orthogonal, since $Q_j^TQ_j = I$ implies $\hat{Q}_j^TM\hat{Q}_j = I$. This is also a property of the eigenvectors X of this generalized eigenproblem. The approximate eigenvectors will eventually be computed in the subspace $span(\hat{\mathcal{Q}})$ regardless of the form used for the Lanczos recurrence. M-orthogonality will introduce difficulties in implementation, but the advantages of performing the recursion in the correct subspace are well documented in [37]. The Lanczos recurrence in this subspace is given in Figure 3.

FIG. 3. *Generalized Symmetric Block Lanczos Algorithm*

Initialization:

Set $\hat{Q}_0 = 0$
Set $B_1 = 0$
Choose R_1 and M-orthonormalize the columns of R_1 to obtain $\hat{Q}_1$ (see §4.1)

Lanczos Loop:

For $j = 1, 2, 3 \ldots$ do
Set $U_j = H\hat{Q}_j - M\hat{Q}_{j-1}B_j^T$
Set $A_j = \hat{Q}_j^T M U_j$
Set $W_{j+1} = U_j - M\hat{Q}_j A_j$
Solve $MR_{j+1} = W_{j+1}$
Compute the M-orthogonal factorization $\hat{Q}_{j+1}B_{j+1} = R_{j+1}$
End loop

Here the matrix M is used at several occasions to assure the M-orthogonality of the Lanczos vectors, i.e., to assure that

$$Q_j^T M Q_j = I.$$

In particular, the last step requires computing the M-orthogonal factorization of R_{j+1}. Standard derivations of the orthogonality of the Lanczos vectors easily generalize to show that these vectors are M-orthogonal, that is, orthogonal with respect to the inner product defined by $x^T My$.

It appears that $M^{-1/2}$ has disappeared from the standard recurrence, only to reappear in disguise as a solution operation. Indeed, (10) applied to the original problem $Kx = \lambda Mx$ is merely an implicit form of the explicit reduction to standard form. This is not the case when H is taken as the operator in the spectral transformation. Substituting $M(K - \sigma M)^{-1}M$ for H gives:

$$M\hat{Q}_{j+1}B_{j+1} = M(K - \sigma M)^{-1}M\hat{Q}_j - M\hat{Q}_j A_j - M\hat{Q}_{j-1}B_j^T. \tag{11}$$

M now appears in *all* of the terms in the recurrence. Formally we can premultiply (11) by M^{-1} to obtain a recurrence

$$\hat{Q}_{j+1}B_{j+1} = (K - \sigma M)^{-1}M\hat{Q}_j - \hat{Q}_j A_j - \hat{Q}_{j-1}B_j^T \tag{12}$$

in which M^{-1} does not appear. This has advantages generally and, in particular, allows us to apply the same recurrence even when M is semidefinite. The justification for doing so appears later in §2.4.

At this point we shall drop the fiction of $\hat{Q}$. All operations will take place in this space, and we shall no longer bother putting 'hats' on the matrices. The actual Lanczos recurrence for solving (4) is given in Figure 4.

FIG. 4. *Block Lanczos Algorithm for the Vibration Problem*

Initialization:

Set $Q_0 = 0$
Set $B_1 = 0$
Choose R_1 and orthonormalize the columns of R_1 to obtain Q_1
with $Q_1^T(MQ_1) = I_p$

Lanczos Loop:

For $j = 1, 2, 3 \ldots$ do
Set $U_j = (K - \sigma M)^{-1}(MQ_j) - Q_{j-1}B_j^T$
Set $A_j = U_j^T(MQ_j)$
Set $R_{j+1} = U_j - Q_jA_j$
Compute Q_{j+1} and (MQ_{j+1}) such that
a) $Q_{j+1}B_{j+1} = R_{j+1}$
b) $Q_{j+1}^T(MQ_{j+1}) = I_p$
End loop

Assuming the matrix (MQ_{j+1}) is actually stored (at least temporarily), the algorithm as written requires only one multiplication by M per step and no factorization of M is required. However, the appropriate implementation of the last step of the Lanczos loop is not as obvious as it may seem. This *M-orthogonalization* of a set of p vectors will be discussed in more detail in §4.1.

Our next goal is to understand how the eigenvalue approximation results for the ordinary Lanczos algorithm generalize to the spectral transformation block Lanczos algorithm. As before, combining all j instances of (12) into one equation yields

$$(K - \sigma M)^{-1}M\mathcal{Q}_j = \mathcal{Q}_jT_j + Q_{j+1}B_{j+1}E_j^T, \tag{13}$$

where $\mathcal{Q}_j$, T_j, and E_j are defined as in (7). Premultiplying (13) by $\mathcal{Q}_j^TM$ and using the M-orthogonality of the Lanczos vectors, it follows that

$$\mathcal{Q}_j^TM(K - \sigma M)^{-1}M\mathcal{Q}_j = T_j.$$

Hence, T_j is the M-orthogonal projection of $(K - \sigma M)^{-1}$ onto the block Krylov subspace spanned by the columns of $\mathcal{Q}_j$. The eigenvalues of T_j will approximate the eigenvalues of (4). If (s, θ) is an eigenpair of T_j, i.e.,

$$T_js = s\theta,$$

then $(y = \mathcal{Q}_js, \theta)$ will be an approximate eigenpair of

$$M(K - \sigma M)^{-1}My = \mu My.$$

However, we are interested in eigenvalue approximations to the original problem (3) and not to the shifted and inverted problem (4). Formula (5) describes the relationship between the spectra of the two problems: if θ is an approximate eigenvalue of T_j, (5) implies that

$$\nu = \sigma + \frac{1}{\theta}$$

is an approximate eigenvalue of (3). Since the spectral transformation does not change the eigenvectors, y is an approximate eigenvector for (3).

The *a posteriori* residual bound (8) does not generalize quite so cleanly. The computation analogous to (8) results in

$$(K - \sigma M)^{-1} M y - y\theta = Q_{j+1} B_{j+1} E_j^T s. \tag{14}$$

For $\theta \neq 0$ it follows that

$$\frac{1}{\theta} M y - (K - \sigma M) y = \frac{1}{\theta} (K - \sigma M) Q_{j+1} B_{j+1} E_j^T s,$$

or

$$(K - \nu M) y = -\frac{1}{\theta} (K - \sigma M) Q_{j+1} B_{j+1} E_j^T s.$$

The quantity on the right is computable without explicitly computing the eigenvector y, but only at the cost of a multiplication by $K - \sigma M$. This is not desirable because $K - \sigma M$ is otherwise not used in the recurrence — only the factors of $K - \sigma M$ are assumed to be available. In §4.2 we present a better way to obtain a residual bound. (Note that $\theta = 0$ corresponds to an infinite eigenvalue of (3), which should not appear in T, as discussed below. Very small θ correspond to eigenvalues far from the shift. The usual convergence results predict that these will be very inaccurate, which is reflected in the division by θ in the residual bounds.)

2.4. Semidefiniteness in the Metric M. Throughout the discussion above, we have made the assumption that M is a positive definite matrix. The important case of a semidefinite M, where there are vectors $z \neq 0$ such that

$$Mz = 0,$$

remains to be considered. The formulation of the block Lanczos algorithm for the vibration problem does not require the factorization of M. Hence the spectral transformation Lanczos algorithm can be applied formally in this case without further modifications. However, the eigenproblem (3) has both finite and infinite eigenvalues and the effect of the infinite eigenvalues on the convergence of the finite eigenvalues is unclear. Fortunately, we need only to make the obvious block modification of the analysis in [28] to remove the infinite eigenpairs from the recurrence. Following Nour-Omid et. al., the starting block for the Lanczos algorithm should be computed as in Figure 5.

Here R_1 is a block of vectors, chosen by some (arbitrary) rule and $Q_1 B_0$ is the M-orthogonal factorization of R_1. We take the M-orthogonal block Q_1 as the starting block for the recurrence.

FIG. 5. *Computation of the Starting Block*

Choose	$\tilde{R}_1$
Compute	$R_1 = (K - \sigma M)^{-1} M \tilde{R}_1$
M-orthogonalize	$R_1 = Q_1 B_0$

The eigenvectors of $Kx = \lambda Mx$ corresponding to finite eigenvalues consist of a component orthogonal to the null vectors of M and a component in the nullspace of M. Ericsson [13] shows that the second, nullspace, component is determined by an algebraic constraint from the non-nullspace component. The constraint expresses the fact that all of these eigenvectors lie in the range of $(K - \sigma M)^{-1} M$. It is shown in [13, 28] that all of the Lanczos vectors lie in this subspace when the starting vectors are chosen in this subspace, as above. The effect of this choice of starting block is that infinite eigenvalues have no influence whatsoever on the block Lanczos algorithm in exact arithmetic. There is the possibility in finite precision arithmetic that infinite eigenvalues reappear in spite of this choice. In §4.2 we add a final postprocessing step to purge the approximate eigenvectors of components not satisfying the constraint.

2.5. A Spectral Transformation for Buckling Problems. The final point to be discussed in this section is the implementation of the spectral transformation for the buckling problem

$$Kx = \lambda K_\delta x, \tag{15}$$

where K is the symmetric positive semidefinite stiffness matrix and K_δ is the symmetric differential or geometric stiffness matrix. Typically only a few eigenvalues closest to 0 are wanted. A simple approach would be to interchange the roles of K and K_δ and to compute the largest eigenvalues of the problem

$$K_\delta x = \mu K x, \tag{16}$$

with $\mu = 1/\lambda$ by applying the simple Lanczos algorithm without shifts. This reciprocal approach has the three drawbacks as (2): it requires the factorization of the possibly semidefinite matrix K, it does not allow for shifting, and the Lanczos vectors would not be in the same subspace as the approximate eigenvectors. However, it is often effective when K is positive definite because the number of eigenvalues sought is rarely large.

An alternative form of the spectral transformation [19] allows more general forms of (15), particularly the semidefinite K case. The operator $K - \sigma K_\delta$ is factored, but the Lanczos recurrence is carried out using K-orthogonality among the Lanczos vectors. This modification is easy to implement by replacing each multiplication by the mass matrix M in the vibration case with a multiplication by the stiffness matrix K in the buckling case; the rest of the recurrence remaining the same. Hence, in the buckling case, the shifted and inverted problem

$$K(K - \sigma K_\delta)^{-1} K x = \mu K x \tag{17}$$

is solved instead of the original problem (15).

In the buckling spectral transformation (λ, x) is an eigenpair of (15) if and only if $(\frac{\lambda}{\lambda-\sigma}, x)$ is an eigenpair of (17). Hence the *buckling spectral transformation* does not change the eigenvectors, and the eigenvalues are related by

$$\mu = \frac{\lambda}{\lambda - \sigma}.$$

These results can be obtained directly, or by applying the vibration spectral transformation with reciprocated shifts to the reciprocal problem (16).

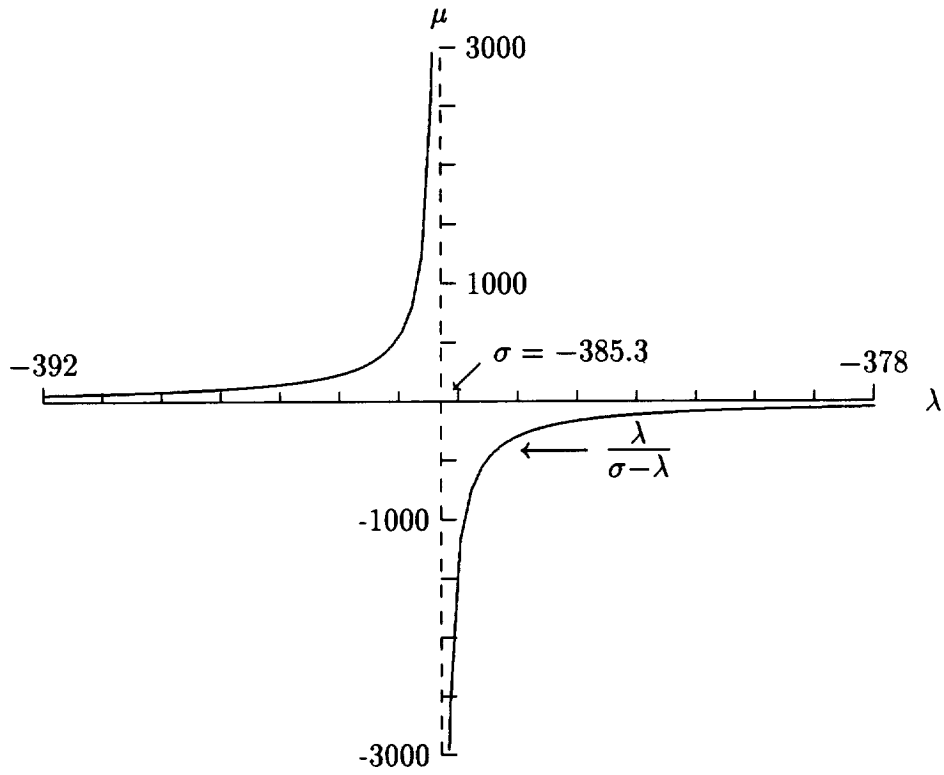
The advantages of the buckling spectral transformation are essentially the same as those of the vibration spectral transformation. Large eigenvalues of the buckling problem are transformed to a cluster of eigenvalues near unity. Eigenvalues near the shift σ are transformed into well separated eigenvalues, which are easily computed by the Lanczos algorithm. The major difference is that a shift at $\sigma = 0$ is not allowed, since all eigenvalues would be transformed to 1. This singularity in the transformation also affects shifts close to zero; very small shifts should not be taken in this form of the transformation. Figure 6 shows an example of the buckling spectral transformation. Table 3 gives details for the eigenproblem BCSST_28, treated as if it were a buckling problem. The initial shift is negative because we ordinarily expect the first negative eigenvalue to be the eigenvalue of most interest in buckling problems (see §3.8). Just as in the case of the vibration spectral transformation, we see that the shift does not need to be close to the desired eigenvalues in any absolute sense. Indeed, in this case the shift is on the wrong side of the origin and yet still has the correct effect on relative separation. Note that because of the scale used to replicate this real example, all the desired eigenvalues are off-scale, and it is not possible to see that the asymptotes for this transformation are at $y = 1$ rather than $y = 0$.

TABLE 3
Buckling Spectral Transformation of BCSST_26, $\sigma_1 = -385.3$

			original		transformed	
i	$\lambda(i)$	$\mu(i)$	gap	relative gap	gap	relative gap
1	4.6×10^{3}	9.23×10^{-1}	6.4×10^{3}	1.2×10^{-11}	4.3×10^{-2}	5.6×10^{-1}
2	1.1×10^{4}	9.66×10^{-1}	2.5×10^{2}	4.6×10^{-13}	7.3×10^{-4}	9.5×10^{-3}
3	1.1×10^{4}	9.67×10^{-1}	2.5×10^{2}	4.6×10^{-13}	7.3×10^{-4}	9.5×10^{-3}
4	1.5×10^{4}	9.74×10^{-1}	3.4×10^{3}	6.3×10^{-12}	7.2×10^{-3}	9.4×10^{-2}
⋮	⋮	⋮	⋮	⋮	⋮	⋮
1919	2.4×10^{14}	1.00×10^{0}	2.9×10^{13}	5.4×10^{-2}	2.3×10^{-13}	3.0×10^{-12}
1920	3.0×10^{14}	1.00×10^{0}	3.6×10^{11}	6.7×10^{-4}	1.6×10^{-15}	2.0×10^{-14}
1921	3.1×10^{14}	1.00×10^{0}	3.6×10^{11}	6.7×10^{-4}	1.6×10^{-15}	2.0×10^{-14}
1922	5.4×10^{14}	1.00×10^{0}	2.4×10^{14}	4.4×10^{-1}	5.5×10^{-13}	7.1×10^{-12}

Except for the different role of the stiffness matrix K, all implementation details are the same for vibration and buckling analysis. Issues involving the M-orthogonality of the Lanczos vectors apply equally to the K-orthogonal Lanczos vectors in the buckling case.

FIG. 6. *Buckling Spectral Transformation*



Since the stiffness matrix K is used in the initialization phase in the same way as M in the vibration case, the sequence of Lanczos vectors will be orthogonal to the space spanned by the eigenvectors corresponding to zero eigenvalues of K. Hence T_j will contain no approximations to the exactly zero eigenvalues of K, which are also zero eigenvalues of (15). This is desirable since typically the first *nonzero* eigenvalue of (15) is what is wanted. The eigenvalues computed by the buckling spectral transformation Lanczos method are distinct from zero.

The eigenvalues of T_j approximate the eigenvalues of (17). Hence, if (s, θ) is an eigenpair of T_j, that is,

$$T_j s = s\theta,$$

then an approximate eigenpair (ν, y) of (15) can be found by

$$\nu = \frac{\sigma\theta}{\theta - 1}$$

and

$$y = Q_j s.$$

These approximate eigenvectors y form a K-orthonormal set. Bounds on the residuals of approximate eigenpairs will be derived in §4.2.

3. A Strategy for Choosing Shifts. The problem to be solved is to find some of the eigenvalues and eigenvectors of

$$KX = MX\Lambda$$

or

$$KX = K_\delta X\Lambda.$$

It is critical to emphasize the fact that we want some, not all, of the eigenvalues. The eigenvector matrix X is almost always a dense matrix even though the original matrices are sparse. We can take advantage of the sparsity of K and M only if we can restrict our attention to a small subset of the eigenvalues.

It is generally sufficient to specify the significant eigenvalue restrictions on the eigenvalues desired by a combination of:

- an ordinal description (e.g., the 50 eigenvalues of smallest magnitude), and
- a physical restriction (e.g., only eigenvalues below 200 or eigenvalues closest to 600).

Both restrictions can be used together, as for example, the 45 lowest eigenvalues in $[10, 100]$. Each can also be used separately, as the smallest 70 eigenvalues, or as all eigenvalues in $[0, 30]$. The problem can be written in its general form as:

- find the p eigenvalues of smallest magnitude in $[a, b]$ and their eigenvectors, or
- find the p eigenvalues of largest magnitude in $[a, b]$ and their eigenvectors, or
- find the p eigenvalues in $[a, b]$ closest to ξ and their eigenvectors, or

- find all eigenvalues and eigenvectors in $[a, b]$.

Here $[a, b]$ is the *computational interval*, which can be finite (both a and b finite), semi-infinite (only one of a and b finite), or infinite (no restrictions at all). Note that the problem of finding the algebraically least eigenvalues in an interval can be transformed into one of finding the eigenvalues of smallest magnitude by a suitable shift of origin.

The purpose of the spectral transformation is to transform the original problem into one whose dominant eigenvalues represent some of the desired eigenvalues. The dominant eigenvalues of the transformed problem correspond to the eigenvalues of the original problem nearest σ.

There are two major goals that drive our strategy for choosing shifts. One is efficiency – we would like to choose a sequence of shifts $\sigma_1, \sigma_2, \ldots, \sigma_s$ so that the total cost, including the cost of the s factorizations and the costs of the individual Lanczos runs, is minimized. This is a difficult problem whose optimal solution depends on the spectrum of the problem, which we cannot know. We have already given an example in Table 1 that shows that there is a better solution than choosing a single shift to compute a very large number of eigenvalues. Our heuristic approach to measuring and reducing cost is described in §3.2 and §4.4.

The second goal of our shift selection is robustness. It is not effective to compute incorrect or incomplete answers, even if done quickly. A paramount objective for our design was a code that would be able to compute all of the desired eigenpairs accurately, except under extreme, pathological, conditions. Further, we wanted a code that could diagnose and report any failures. This emphasis on robustness separates our code from the Ericsson-Ruhe shifting scheme [12, 15], which may detect failures, but is unable to correct poor choices of shifts. Our emphasis on robustness without user intervention distinguishes us from the code described in [9], which attempts to solve the much harder general eigenvalue problem. The tools we use to create robustness, trust intervals and matrix inertias, are an appropriate place to begin the detailed discussion of our choices of shifts.

3.1. Trust Intervals, Matrix Factorizations and Inertias. Suppose that during the course of eigenanalysis, we have computed a set of eigenvalues lying between two shifts σ_1 and σ_2. We would like to confirm whether these are, in fact, all the eigenvalues in this interval. The key tool for sparse eigenvalue computation is the matrix inertia, as computed from a factorization of a shifted matrix. Suppose that C is a real symmetric matrix, which has been decomposed as

$$C = LDL^T.$$

The *inertia* of C is the triple (π, ν, ζ) of integers, where π is the number of positive eigenvalues, ν the number of negative eigenvalues and ζ the number of zero eigenvalues.

Sylvester's Inertia Theorem ([31],p. 10) states that the inertia of a matrix is invariant under congruence, that is, for all nonsingular matrices F, the inertia of F^TCF is the same as that of C. Note that $D = L^{-1}\ C\ L^{-T}$, that is, D is congruent to C (with $F = L^{-T}$). By Sylvester's theorem, the number of negative entries in D is the number of negative eigenvalues from C. (The entries in D are not eigenvalues – only the signs are matched.) It is an easy extension to show that the number of negative eigenvalues of $C - \sigma I$ is the same as the number of eigenvalues of C that are smaller than σ. Thus, the number of negative

terms in D from the LDL^T decomposition of $C - \sigma I$ gives the number of eigenvalues smaller than σ. Frequently, in an abuse of notation, $\nu(C - \sigma I)$, is referred to as the inertia of the shifted matrix; it is also called the Sturm sequence number in engineering references.

It is easy to see that $\nu(C - \sigma_2 I) - \nu(C - \sigma_1 I)$ is the number of eigenvalues in the interval $[\sigma_1, \sigma_2]$ (assuming $\sigma_1 < \sigma_2$ and the two factorizations are nonsingular). Thus, the factorizations that are taken at σ_1 and σ_2 provide the inertias at these two shifts; from the inertias we determine the number of eigenvalues in the interval $[\sigma_1, \sigma_2]$. We can now compare the number of eigenvalues expected in the interval with the number actually computed. When these numbers agree, we say that the interval $[\sigma_1, \sigma_2]$ is a *trust interval*, trusted because we have computed precisely the number of eigenvalues that we should have computed. Our shifting strategy is driven by the goal of establishing a trust interval around all of the desired eigenvalues.

However, applying these Sturm sequence results to generalized eigenproblems requires a transformation from the ordinary eigenvalue problem $CX = X\Lambda$ to the generalized problem $KX = MX\Lambda$. These generalizations are not entirely straight-forward. We present the results here; proofs are found in Appendix A. We compute $K - \sigma M = LDL^T$ (or $K - \sigma K_\delta = LDL^T$), and we want to draw conclusions from $\nu(K - \sigma M) = \nu(LDL^T)$. The interpretation of $\nu(LDL^T)$ when these generalized symmetric eigenproblems are well-posed is given in Figure 4. The major surprise in this table of the appearance of a second term when the matrix used as a norm is only a seminorm. This term corresponds to an assignment of signs to the infinite eigenvalues in the vibration case and the zero eigenvalues in the buckling case. We note here that the term $\dim(\mathcal{N}(M))$ in the vibration case does not appear if K is positive semidefinite, typically the case in structural engineering practice. If it does appear, it adds a complication to the problem of finding the algebraically smallest eigenvalues, because the infinite eigenvalues are the algebraically smallest eigenvalues. It makes the problem of finding the eigenvalues of smallest magnitude slightly more difficult.

FIG. 7. *Interpretation of* $\nu(K - \sigma M)$ *or* $\nu(K - \sigma K_\delta)$

vibration analysis:	
M positive definite	# of eigenvalues $< \sigma$
M positive semidefinite	(# of eigenvalues $< \sigma$) $+ \gamma$
	$\gamma = \begin{cases} 0 & \text{some cases} \\ \dim(\mathcal{N}(M)) & \text{other cases} \end{cases}$
buckling analysis:	
K positive definite	# of eigenvalues in $(0, \sigma)$ or $(\sigma, 0)$
K positive semidefinite	(# of eigenvalues in $(0, \sigma)$ or $(\sigma, 0)$) $+ \gamma$
	$\gamma = \begin{cases} 0 & \sigma \text{ of one sign} \\ \dim(\mathcal{N}(K)) & \sigma \text{ of other sign} \end{cases}$

The term $\dim(\mathcal{N}(K))$ in buckling analysis is more significant. The usual problem of finding the eigenvalues of smallest magnitude becomes impossible because the zero eigenvalues cannot be computed directly. The problem still can be solved if $\dim(\mathcal{N}(K))$ is known, either adventitiously or by being computed as a partial eigenanalysis of K. The problem of

finding the eigenvalues of smallest magnitude in an interval bounded away from zero is still well-posed.

The result of a successful eigenextraction is a trust interval containing all of the desired eigenvalues. The eigenvalues desired will be all of the eigenvalues in a subinterval of the computational interval. In general we will have to compute the inertia (and the factorization) at each end point of the subinterval. Each factorization typically represents a considerable fraction of the cost of a block Lanczos run for the corresponding shifted operator. Therefore, it is worthwhile to combine, wherever possible, the shifts for verifying eigenvalue counts, and the shifts for Lanczos runs. The design of the shift selection approaches this goal by creating as soon as possible a trust interval containing some of the desired modes, and thereafter, extending that trust interval to contain more, and eventually all, of the desired modes.

The process begins with an initial shift at some point σ_1. The factorization is followed by a Lanczos run with the shifted operator $(K - \sigma_1 M)^{-1} M$ (or its counterpart in buckling analysis). At the conclusion of the Lanczos run, some eigenvalues and eigenvectors are returned. In all cases the first trust interval has been computed, because $[\sigma_1, \sigma_1]$ is an empty trust interval. A larger trust interval may be available to us because often $\nu(K - \sigma_1 M)$ gives the number of eigenvalues in $(-\infty, \sigma_1)$ (in buckling analysis, either $(0, \sigma_1)$ or $(\sigma_1, 0)$). Should the Lanczos run have computed that many eigenvalues to the left of σ_1, the first trust interval becomes nontrivial. It is not unusual for this to occur, but it is unusual for this first trust interval to contain all the requested eigenvalues.

In general, it will be necessary to take a second factorization, at σ_2, at least to provide a second inertia. The first decision that has to be made is whether σ_2 should be to the left or to the right of σ_1. We choose a direction from the eigenvalues converged during the Lanczos run at σ_1, the inertia at σ_1 and the description of which eigenvalues were requested. In the following we assume expansion of the trust interval to the right, but the actual algorithm can and does go both ways. If only some of the desired eigenvalues were computed during the first Lanczos run, we would like to make the factorization at σ_2 serve both as a basis for an inertia computation and as the factorization for a new Lanczos run. We would like to take σ_2 close enough to σ_1 so that at the end of the second Lanczos run $[\sigma_1, \sigma_2]$ will form a trust interval – all the eigenvalues in the subinterval will have been computed by one of the two Lanczos runs. We proceed to expand the subinterval $[\sigma_1, \sigma_2]$ to include more of the desired eigenvalues.

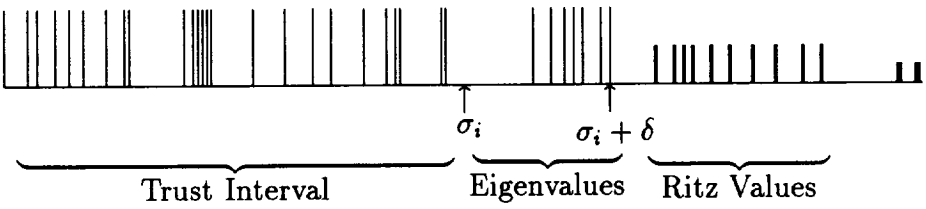
The selection of σ_2 is complicated by the desire to be efficient in the use of the factorization and of the block Lanczos algorithm. Ideally we would choose σ_2 close enough to σ_1 that the second Lanczos run finds all the remaining eigenvalues in the interval, and at the same time, we would like σ_2 to be far enough away from σ_1 so that the second Lanczos run stops, for efficiency reasons, exactly when it has computed all the missing eigenvalues. Thus, a simple description of our shift selection is that we choose each new shift to *maximally* extend an existing trust interval.

Obviously the desire to maximally extend the trust interval presents a conflict between the goal of maintaining trust and the goal of extending as far as possible. Resolving this conflict is the topic of the next section.

3.2. Shifting to Extend a Trust Interval. Successive shifts are taken to extend the initial trust interval containing σ_1. Each new shift, as discussed, should attempt to maximally extend the trust interval. Unfortunately we do not know exactly where the uncomputed eigenvalues lie. After the initial shift and Lanczos run we do have some information available on which to base a selection, including any computed eigenvalues, other knowledge about the existing trust interval, and additional information from the previous Lanczos runs.

In general, each Lanczos run creates a set of approximations to eigenvalues. Some of these meet our criterion for accuracy. For simplicity, we refer to these as eigenvalues, although they are really accurate approximations thereto. The remainder of the approximations created by the Lanczos algorithm are not (yet) acceptable as eigenvalues. They may be close to eigenvalues or they may be poor approximations to eigenvalues, but they do provide a general picture of the distribution of the uncomputed eigenvalues. Figure 7 gives an illustration of the general situation, in which the last Lanczos run was at a shift σ_i that forms the right end point of a trust interval. The tallest, thin, lines denote eigenvalues. The lines of medium height and width are approximations which are not yet acceptable as eigenvalues, but which do have accuracy estimates good enough to know that at least one significant digit is correct. We call these *Ritz values.* (The Lanczos approximations are Ritz values, but we abuse the mathematical term to describe only those approximations that are not good enough to be accepted, and not bad enough to be meaningless.) The short, broad, lines denote approximations whose accuracy estimates are larger than their values.

FIG. 8. *Trust Intervals*



The shift selection assumes that the inverted spectrum as viewed from σ_{i+1} will be similar to the inverted spectrum as viewed from σ_i. One view of this similarity of inverted spectra is that if the Lanczos run from σ_i computed k eigenvalues to the right of σ_i efficiently, we expect that an efficient run at any σ_{i+1} should compute k eigenvalues to its left. Thus σ_{i+1} should be placed between the $2k - th$ and $(2k+1) - st$ eigenvalues to the right of σ_i. We only know k of these eigenvalues, which we assume are the first k. We use the first k Ritz values to estimate the missing eigenvalues, and place the new shift σ_{i+1} between the $k - th$ and $(k+1) - st$ Ritz values. It is desirable that the shift not be extremely close to an eigenvalue — if it were, the eigenvalue near the shift would completely dominant the Lanczos recurrence. For that reason we choose the bisector of the two specified Ritz values as the shift, rather than using a Ritz value itself as a shift. Further, we use a relaxed tolerance to detect "clusters" of eigenvalues, and bisect clusters rather than Ritz values.

Unfortunately, it is possible that there are fewer than k Ritz values available to the right of σ_i, or perhaps none at all. This situation will arise if σ_i were chosen extremely close to an eigenvalue, in which case the Lanczos approximations to the nearby eigenvalue would be accurate, but none of the Ritz values would be accurate enough to specify the new shift. This seemingly unlikely case is particularly likely to happen when K is semidefinite and the first shift is taken at zero. Zero is a reasonable choice, yet is also an eigenvalue.

To treat such cases we use a second view of the inverted spectra, based on the assumption that the "radius of convergence" should be about the same for each shift. We define δ to be the maximum of its previous value and the distance between the right end point of the trust interval and the rightmost computed eigenvalue (see Figure 7). Initially, δ is set to the problem scale (see §3.4). Then a second choice for the next shift is $\sigma_{i+1} = \sigma_i + 2 * \delta$.

We have to choose between these two possible shifts. We take the more aggressive choice, the maximum of the two possibilities, in the case where we still need to compute more eigenvalues than we have knowledge of Ritz values. If more Ritz values are available than there are eigenvalues left to compute, we choose the next shift based solely on the Ritz values, ignoring the shift based on δ.

We have discussed these general rules for choosing σ_{i+1} in the case where σ_{i+1} is to the right of σ_i. Of course, we obtain two similar views of the spectra to the left of σ_i, which give another alternative for the next shift. In general we do not know in which the direction the next shift should be. Indeed, there are circumstances in which we first move in one direction from σ_i and then in the other direction. This occurs when we find eigenvalues nearest an interior point of the computational interval or when we need to go backwards to find missing eigenvalues. However, we do not care to deal at any time with all the Ritz values from all of the shifts; interpreting this possibly redundant set of values would be quite complicated. For that reason, at the completion of each Lanczos run in which we attempted to extend a trust interval, we compute, and save, the next shift which would extend the new trust interval further in the same direction. The first shift, unless it is at a finite end point of the computational interval, is treated as extending the null trust interval both to the left and to the right. The Ritz values can be discarded after the tentative shift has been computed.

These two views of the inverted spectra, albeit simplistic, have proven to be effective. The assumption that the two inverted spectra will have similar convergence behavior is a gross simplification, but we have been unable to develop a model that performs better in practice. Ideally we would use the Ritz values to predict the overall convergence rates of the eigenvalues, but the best current model for the convergence of interior eigenvalues [35] is far too pessimistic to be of any use here.

There are cases where one or more additional factorizations are required simply to establish the trustworthiness of an interval. As a simple and common example, assume that the ten least eigenvalues are required in a vibration analysis. An initial shift is chosen as described in the next section, and a single Lanczos run is made with this shift. The Lanczos run computes ten eigenvalues. If these ten eigenvalues are in fact the least ten eigenvalues, this can be verified by computing the inertia for a shift lying between the tenth and eleventh eigenvalues. In practice, the bisector of the tenth computed eigenvalue and the Ritz value that we believe estimates the eleventh eigenvalue is taken as σ_2. If $\nu(K - \sigma_2 M) = 10$ the

eigenproblem has been solved. If not, a special case of missing eigenvalues must be handled (see §3.7.1).

3.3. Sentinels. There are several aspects of our eigenanalysis code where the shift selection mechanism and the implementation of the Lanczos algorithm are closely tied together. For example, we do not want to recompute at later shifts eigenpairs that have been computed from earlier shifts. Any computation spent recomputing known eigenpairs is wasted. Even allowing accidental recomputation creates a difficult situation in which we must determine the correct multiplicity of a computed eigenvalue for which several eigenvectors have been computed. We choose never to allow this situation to arise.

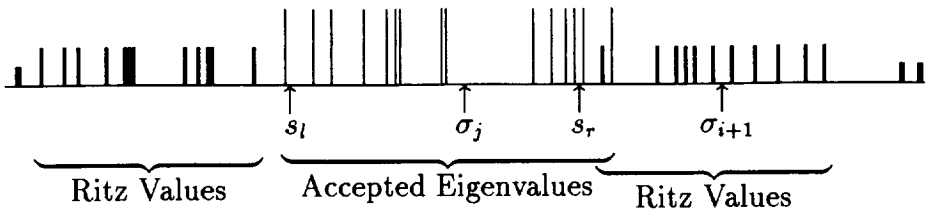
In theory there is a very simple fix. If the starting block for the Lanczos recurrence is chosen to be M-orthogonal to all previously computed eigenvectors, the recurrence should remain M-orthogonal to all previously computed eigenvectors. This is not sufficient in practice, as rounding errors introduce components of the excluded eigenvectors. One could, at each step, reorthogonalize the Lanczos blocks to the excluded eigenvectors, but that is unnecessary. A better solution is to reorthogonalize the recurrence to specific eigenvectors only when necessary; the mechanism for doing so is *external selective orthogonalization*, the topic of §4.3.3. This mechanism dramatically reduces the cost of preventing the reappearance of excluded eigenvectors.

A second mechanism for reducing this cost is in the purview of the shifting code. A common situation is depicted below, in Figure 8. The new shift, σ_{i+1}, has been chosen; the nearest previous shift, σ_j, forms the end of a trust interval. (Figure 8 depicts the initial case where the trust interval including σ_j is trivial.) Between the two shifts lie a set of eigenvalues and Ritz values computed during the run at σ_j. Because the convergence rate for the eigenvalues in the Lanczos algorithm decreases as the distance from the shift increases (as one moves into the interior of the shifted and inverted spectrum), the usual pattern is that the accepted eigenvalues are those closest to σ_j and the Ritz values are those farther out. There is usually no interlacing of the two sets. It is also typical that the Lanczos algorithm does not miss eigenvalues – an eigenvalue near the shift will be represented as some sort of Ritz value. (Eigenvalues of high multiplicity may be represented by fewer Ritz values than the multiplicity warrants.) We expect that any eigenvalues not yet computed between σ_j and σ_{i+1} will be found in the region represented by the (unaccepted) Ritz values as opposed to the region in which eigenvalues converged.

Consider in each direction the eigenvalue farthest from σ_j such that between it and σ_j no (unaccepted) Ritz values are found. There is such an eigenvalue to the right of a shift and similarly to the left, each being the last eigenvalue before a Ritz value is found. We call these two eigenvalues λ_r^* and λ_l^*. In normal circumstances we assume that there are no eigenvalues missing between σ_j and λ_r^* or λ_l^*.

We define the *right sentinel* s_r as the left end point of the interval of uncertainty for λ_r^*, based on the required accuracy tolerance. We know that the true value of λ_r^* lies to the right of the sentinel s_r. A *left sentinel* is defined similarly. We perform the Lanczos run at the new shift σ_{i+1} using limited orthogonalization. Assume $\sigma_{i+1} > \sigma_j$. The eigenvectors corresponding to λ_r^* and to any other eigenvalues found between s_r and σ_{i+1} are prevented from reappearing by use of external selective orthogonalization. We allow the recurrence to

FIG. 9. *Sentinels*



recompute eigenvalues which lie to the left of s_r, but these are discarded immediately. This technique allows us to trust any eigenpairs that are computed in the region where we expect new eigenpairs to appear, without our incurring the cost of extensive reorthogonalization. The reorthogonalization with respect to λ_r^*'s eigenvector removes any doubt that could exist about the exact location of this eigenvalue in the shifted and inverted spectrum for the new shift. At the same time, the eigenvector(s) most likely to reappear are suppressed.

We generalize the notion of sentinels slightly to handle clusters of eigenvalues. Should the sentinel s_r lie to the left of λ_{r-1}, we move the sentinel back to the end point of the uncertainty interval for λ_{r-1}^*. We continue this process until the sentinel lies between the intervals of uncertainty for two eigenvalues, or until the shift itself is used as the sentinel.

3.4. The Initial Shift. The most difficult task is usually the first: getting started. The selection of the first shift must be made with no information about the spectrum other than the specification of the desired eigenvalues. We use any location information in the specification to make an initial choice for the first shift, σ_1.

$$\sigma_1 = \begin{cases} \left.\begin{array}{ll} a & \text{if } |a| \le |b| \\ b & \text{if } |a| > |b| \end{array}\right\} & \text{if lowest modes or all modes wanted and } \min|a|,|b| < \infty \\ \left.\begin{array}{ll} a & \text{if } |a| \ge |b| \\ b & \text{if } |a| < |b| \end{array}\right\} & \text{if highest modes wanted and } \max|a|,|b| < \infty \\ \left.\begin{array}{ll} a & \text{if } |a| < |b| \\ b & \text{if } |a| > |b| \end{array}\right\} & \text{if highest modes wanted and } \min|a|,|b| < \max|a|,|b| = \infty \\ \xi & \text{if modes nearest } \xi \text{ wanted} \\ 0 & \text{otherwise} \end{cases}$$

This choice of σ_1 gives a reference point in the spectrum as to which eigenvalues are important to the user. In cases where ξ is not specified by the user, we define ξ to be σ_1 as defined above. We note that 0 is a natural choice when we have no location information — in that common case we want the eigenvalues of least magnitude, i.e., closest to 0.

Unfortunately a choice of $\sigma_1 = 0$ is fraught with difficulties. A shift at zero is not allowed in the buckling transformation and yields a singular operator in vibration analysis when K is semidefinite. If a shift at zero were taken in the latter case, it is unlikely that the singularity of the operator would be detected. It is more likely that only the zero eigenvalues would be computed and no other useful information could be extracted from the run. (The

near-singularity of the operator would cause the Lanczos recurrence to break down after computing the invariant subspace of the zero eigenvalues.) This would leave us little better off than we began, with no information as to where the nonzero eigenvalues are located. A better initial shift would be a shift somewhere in the vicinity of the first few nonzero eigenvalues. Such a shift would allow computing both the zero, rigid body, modes and a number of the nonzero modes as well.

The difficulty is in getting some idea of the scale of the first nonzero eigenvalues. We have adopted a heuristic strategy recommended by Louis Komzsik of The MacNeal–Schwendler Corporation. This heuristic computes the geometric mean of the centers of the Gershgorin circles while excluding the centers smaller than 10^{-4}. This heuristic usually computes a reasonable *problem scale* χ. Specifically,

$$\chi = \frac{1}{\sqrt{n} * \sum \frac{|m_{ii}|}{|k_{ii}|}}$$

where the summation is taken over all $k_{ii} \neq 0, \frac{|m_{ii}|}{|k_{ii}|} < 10^4$. Table 5 gives an idea of the reliability of this heuristic.

We use χ to correct the initial selection of σ_1 whenever $|\sigma_1| < \chi$. In either the vibration problem or ordinary eigenvalue problem we adjust σ_1 as

$$\sigma_1 = \begin{cases} \chi & \text{if } a \leq \chi \leq b \\ -\chi & \text{otherwise, if } a \leq -\chi \leq b \\ \max(|a|, |b|) & \text{otherwise} \end{cases}$$

We adjust the initial shift in a similar fashion for the buckling problem. However, we try $\sigma_1 = -\chi$ first and then $\sigma_1 = \chi$ second, because the most common buckling analysis in structural analysis is an analysis involving the smallest negative eigenvalue.

3.5. Choosing a Direction in which to Expand a Trust Interval. The majority of vibration analyses result in a simple, monotonic, expansion of the trust interval from lowest to higher values. In these cases we know that there are no additional eigenvalues of interest to the left of the trust interval; extending the interval to the right is the only reasonable action. There are other cases in which we have a decision to make. Such cases arise when a shift is taken in the interior of the spectrum by accident or by design. For example, ξ is a very reasonable initial shift when we want to find eigenvalues nearest ξ. In general, finding the eigenvalues of smallest magnitude for an ordinary eigenproblem or for buckling analysis is also such a case.

The reference value ξ, either as set in the problem description or from the initial shift (see §3.4), is used to determine the direction in which to move the shift. If multiple trust intervals exist, the trust interval including or closest to ξ is *primary*; §3.7.1 describes how multiple trust intervals can exist and the logic for determining a new shift in that case. In the most typical case we have only a single trust interval, which we attempt to extend.

We distinguish two subcases, when the trust interval includes an endpoint of the computational interval and not. In the first case the trust interval can only be extended in one direction without moving outside the computational interval, so the choice of direction is

TABLE 4
Comparison of Problem Scale χ and Lowest Eigenvalues

matrix	χ	lowest eigenvalue	closest to eigenvalue
BCSST_08	1.8×10^{-2}	6.9×10^{0}	1
BCSST_09	1.3×10^{7}	2.9×10^{7}	1
BCSST_10	3.1×10^{-3}	7.9×10^{-2}	1
BCSST_11	3.0×10^{2}	1.1×10^{1}	12
BCSST_12	1.5×10^{3}	3.5×10^{3}	1
BCSST_13	1.2×10^{2}	1.5×10^{3}	1
BCSST_19	6.6×10^{0}	2.1×10^{0}	3
BCSST_20	5.5×10^{2}	6.6×10^{0}	7
BCSST_21	1.5×10^{4}	5.1×10^{4}	1
BCSST_22	2.4×10^{4}	7.3×10^{4}	1
BCSST_23	4.0×10^{-1}	4.5×10^{1}	1
BCSST_24	1.4×10^{-1}	4.3×10^{0}	1
BCSST_25	5.3×10^{-8}	9.6×10^{-4}	1
BCSST_27	2.6×10^{-3}	3.1×10^{0}	1
LUND	2.1×10^{1}	2.1×10^{2}	1
PLAT362	2.0×10^{-5}	3.6×10^{-12}	76
PLAT1919	2.1×10^{-6}	1.1×10^{-13}	315

trivial. When the trust interval includes neither endpoint, we further distinguish between cases where ξ is or is not in the trust interval. If the trust interval does not include ξ, we shift in the direction of ξ, because that is where the eigenvalues of most importance to the user lie.

The only remaining case is of a single trust interval that contains ξ, but neither endpoint of the computational interval. In this case we compute the interval $[z_l, z_r]$ that includes the entire trust interval and all computed eigenvalues, even those outside of the trust interval. We define $r = min(\xi - z_l, z_r - \xi)$ to be the radius of a symmetric *umbrella* about ξ where we have some degree of confidence that we have computed all the eigenvalues in the umbrella. Note that this may be an extension of the trust interval and confidence may not be confirmed by inertia values. Our goal is to enlarge this umbrella enough to include all of the eigenvalues that the user has requested or until one end of the umbrella is an end point of the computational interval. We move in whichever direction increases r, i.e. the direction which increases the smaller of $\xi - z_l$ and $z_r - \xi$. Ties are broken by shifting to the left.

3.6. Analysis in a Finite Interval. Frequently the user of the sparse eigensolver will specify a computational interval with finite end points. The number of eigenvalues in the interval is usually valuable information to the user and the eigenanalysis code, even when not all of these eigenvalues are actually computed. We obtain this information by computing two factorizations, one for each end point. If these factorizations can be used in the eigenanalysis itself, the cost of gaining this information would be nominal.

As discussed in the previous section, we often choose the initial shift to be one of the end points. If so, one of the factorizations is required in any case. The factorization at the other end point will be required for situations where all eigenvalues in the interval are desired. Even when not, this factorization will be useful if the number of eigenvalues in the interval is not much greater than the number desired.

We designed our code to compute the factorizations at both end points whenever both end points are finite. These factorizations are saved off-line. We attempt to use these factorizations whenever it appears to be appropriate. The initial shift will often make this quite natural. Furthermore, when the natural choice of a shift would be near an otherwise unused finite end point and when a shift at the finite end point would not cause a large number of extra eigenvalues to be computed, we perturb the choice of shift to use the end point. This may result in some additional work during the Lanczos iteration, but will save the cost of a factorization.

We note that there are cases where we can extend a trust interval to a finite end point without making a Lanczos run at the end point. These occur when the analysis at another shift results in all of the eigenvalues between the shift and the end point.

3.7. Special Cases. Robustness is one of our goals. It is naive to expect that the heuristics described above will work for all problems. Here we describe a number of special cases that can and do arise in practice, and our approaches for handling them smoothly.

3.7.1. Filling Gaps. The shift selection is designed to extend the trust interval obtained from previous Lanczos runs. Strange, asymmetric, distributions of eigenvalues or very high multiplicities may, however, create situations in which the shift σ_{i+1} to extend the trust

interval is taken too far from σ_i to allow computing all the eigenvalues in (σ_i, σ_{i+1}) with a single run. In such a case, the inertias from σ_i and σ_{i+1} will indicate that some eigenvalues between the two shifts have not been computed.

Our heuristic is driven by the goal of maintaining a trust interval. We will find the missing eigenvalues *before* we attempt to extend our knowledge beyond σ_{i+1}. Thus, we attempt to fill the *gap* between the two *active* shifts σ_i and σ_{i+1}, before proceeding.

When eigenvalues are missing between two shifts, we assume that the missing eigenvalues lie between the right sentinel s_i for the shift σ_i at the left and the left sentinel s_{i+1} for the shift σ_{i+1} at the right. These sentinels become the end points of an interval in which we want to compute eigenvalues. In some cases the sentinels define an empty interval; when this occurs we extend the end points of the interval to the nearby end points of the trust intervals bounding the gap. In either case we have a new interval

$$[c, d] = \begin{cases} [s_i, s_{i+1}] & \text{if this is nontrivial} \\ [\sigma_i, \sigma_{i+1}] & \text{otherwise} \end{cases}$$

in which we want to chose a shift. We choose σ_{i+2} as

$$\sigma_{i+2} = \begin{cases} \sqrt{cd} & \text{if } 0 < 2c < d \\ -\sqrt{cd} & \text{if } d < 2c < 0 \\ \frac{c+d}{2} & \text{otherwise} \end{cases}$$

It is not always the case that the gap between two trust intervals is filled on the first attempt. The shifting strategy will continue recursively in its attempt to compute the eigenvalues missing between the primary trust interval (for which σ_i is an endpoint) and its nearest neighbor. We continue in the mode of filling a gap until the primary trust interval has grown large enough to contain the requested eigenvalues or when all trust intervals have been merged into one.

3.7.2. Restart at the Same Shift. Economizing on the number of factorizations is also a goal. There are certain circumstances where the knowledge to be gained from a particular shift is not exhausted by a single Lanczos run. In two such cases we initiate more than one Lanczos run with the same shift.

One case occurs when there are eigenvalues of very high multiplicity. The block Lanczos algorithm may not compute the full multiplicities of eigenvalues when the multiplicity is larger than the block size. We maintain information of the sizes of the clusters of eigenvalues that are computed by the Lanczos algorithm. When a Lanczos run encounters an eigenvalue with multiplicity greater than or equal to the block size and when there are eigenvalues still to be computed, we make an additional Lanczos run at the same shift. During this run we perform external selective reorthogonalization against all the newly computed eigenvectors and any other eigenvectors in the interval between this shift and any neighboring shifts. Our goal is to compute any of the possible missing copies of the eigenvalues with high multiplicity immediately. Note that we discard any use of sentinels because the assumption behind them has probably broken down in the presence of high multiplicity.

A second case occurs when the shift strategy has made an unfortunate shift very close to a true eigenvalue. In this case the block Lanczos procedure will terminate early, with

a rank-deficient residual block. Assuming that more eigenvalues near this shift are still of interest to us, we rerun Lanczos at the same shift, while adding all the newly computed eigenvalues and eigenvectors to the set of vectors for external selective orthogonalization. This may keep the newly computed eigenvalues from dominating the inverted spectrum, thereby allowing the possible computation of more eigenvalues at this shift. Suppression of the dominant eigenvalues is not often effective numerically, but the cost of discovering this is very little. A particular case in which this approach is effective is when the nearby eigenvalue has very high multiplicity, that is, when both of these cases appear simultaneously. This is a particularly difficult situation for the shifting heuristic, making this special case somewhat more likely to occur.

3.7.3. Hole in the Spectrum. Another particularly difficult spectrum for our selection of shifts occurs when there is a very large disparity in the magnitudes of the desired eigenvalues. In such cases our notion of a reasonable distance may be faulty and yet we may have no Ritz value information to help us choose a new shift. There are two simple ways to create such cases. One occurs if we ask to compute eigenvalues larger than the largest finite eigenvalue in a case where M is singular. The second occurs when K is singular and the initial shift is much smaller than the first non-zero eigenvalue; the zero eigenvalues are computed as very small non-zero values, whose magnitude gives us no information about the magnitudes of the other eigenvalues.

Our code treats as special a situation in which no new information is obtained at consecutive shifts. That is, we compute no meaningful Ritz values and the inertias at the two shifts σ_i and σ_{i+1} are identical. We suspect that there is a "hole" in the spectrum, that the remaining eigenvalues are farther away than our notion of a reasonable distance. We expand the notion of a reasonable distance in an attempt to cross the hole. If the computational interval $[a, b]$ has a finite end point that has not been used previously as a shift (see §3.6), the shift strategy will select the new shift at that end point. Otherwise, assuming that we are expanding a trust interval to the right, we take the new shift $\sigma_{i+2} = \sigma_{i+1} + 10\delta$ (see §3.2 for a description of δ). If this Lanczos run still provides no new information, we take $\sigma_{i+3} = \sigma_{i+2} + 100\delta$. If we still obtain no new information, we make a final attempt to cross the gap will be made with a shift $\sigma_{i+4} = \sigma_{i+3} + 1000\delta$. If this run still provides no new information, we terminate on the assumption that the remaining eigenvalues are infinite. We return the eigenvalues already computed, together with an appropriate warning.

3.7.4. Treatment of δ in No Ritz Value Cases. The setting of the "reasonable distance" value, δ, must be made carefully in cases in which the Lanczos algorithm terminates abnormally. This value is not updated if no new information is available for the next shift. A likely cause is a shift that matches an eigenvalue. The particular case of rigid body modes and a shift at zero is handled in this manner, so the reasonable distance value for the second shift is the estimate of the non-zero eigenvalue discussed previously.

3.7.5. Overly Aggressive Shifts. Unusual distributions of eigenvalues or unusual convergence patterns may cause situations in which a shift is selected much farther out than required for the desired eigenvalues. This will occur only when one end of a trust interval is being extended. We determine that the shift is too far from the current trust interval if a run

at this shift will have to compute more than 30 eigenvalues before computing eigenvalues of interest to us. That is, if we need to compute q eigenvalues beyond the trust interval, but there are more than $q+30$ eigenvalues between the shift and the trust interval, the shift has been taken too far out. In such a case we record the old shift, to keep another shift from going out too far in that direction, and select a new shift. The number 30 is a heuristic estimate of the number of eigenvalues we can profitably find with a single run. We choose the new shift by linear interpolation between the end of the trust interval σ_t, and the shift we reject, σ_r. The new shift is:

$$\sigma = \sigma_t + \frac{q}{[\nu(K-\sigma_r M) - \nu(K-\sigma_t M)]}(\sigma_r - \sigma_t).$$

3.8. Modifications for Buckling Problems. The spectral transformation used in the buckling problem for the Lanczos iteration is ill-posed for shifts at or near zero. The shift strategy for the buckling problem is similar to the vibration strategy except that shifts at zero are not allowed. A shift at zero is replaced by one half the minimum of the problem scale χ, the absolute value of the shift nearest to zero, and the absolute value of the computed eigenvalue nearest to zero.

4. Implementation of The Block Lanczos Algorithm. The underpinning of our eigenanalysis code is the block Lanczos algorithm, as specialized for the spectral transformations (§2.3 and §2.5). The use of the block Lanczos algorithm in the context of the spectral transformation and within applications code necessitates careful attention to a series of details: the implications of M-orthogonality of blocks; block generalizations of single vector orthogonalization schemes; effect of the spectral transformation on orthogonality loss; and interactions between the Lanczos algorithm and the shifting strategy. All of these issues are important, but none of them has previously been addressed in detail. The success of the algorithm hinges on their implementation.

4.1. The M-Orthogonal QR Factorization. Each step of the block Lanczos recurrence generates an $n \times p$ matrix R, whose column vectors are to be orthogonalized with respect to an inner product defined by a positive definite matrix. In the two standard engineering analyses the inner product matrix is the mass matrix M in vibration analysis or the stiffness matrix K in buckling analysis. The vibration problem is the most common form; we will let M stand generically for the matrix inducing the norm in either of these problems. We hope that this will not confuse the reader in understanding the algorithm as applied to buckling problems.

Given R, we must compute its orthogonal decomposition QB such that

- $R = QB$
- $Q^T M Q = I$
- Q is $n \times p$
- B is $p \times p$ and upper triangular.

Were M the identity, we would have a number of good choices for computing an orthogonal factorization. When M is not the identity, the choices appear to be limited. There are straightforward modifications to the usual ordinary or modified Gram-Schmidt processes to

account for M. However, both require computation of innerproducts $q_i^T M\hat{q}_j, j < i$, where $\hat{q}_j$ indicates that $\hat{q}_j$ is a vector that has already been modified during the algorithm. In addition, the normalization condition requires $\hat{q}_j^T M\hat{q}_j$.

Multiplication of a vector by M may be expensive in structural engineering applications. We expect M to be sparse, but not necessarily diagonal. Of greater consequence is the fact that M may not be stored in main memory; multiplication by M may require accessing M on secondary storage. For these reasons, we have adopted a generalization of the modified Gram-Schmidt process that requires only matrix-block products, never matrix-vector products. We realize this goal by saving a set of p auxiliary vectors that represent the product MQ throughout the process. This matrix is initialized to MR when the matrix that will hold Q is initialized to R; thereafter, updates made to vectors in Q are shadowed by identical updates in MQ. As a result, M is used explicitly only in the initialization.

This way of enforcing M-orthogonality certainly suggests questions of numerical stability. However, it is well known that the Gram-Schmidt process itself does not guarantee orthogonality of the vectors in a single sweep. Following [10], we repeat the orthogonalization process up to $2p$ times, another repetition being required whenever the norm of any of the q_j vectors is less than η times its norm at the beginning of the iteration. At each repetition the matrix MQ is recomputed by an explicit multiplication by M. The choice of $\eta = \sqrt{2}/2$ from [10] guarantees that the final set of vectors is orthonormal.

In our algorithm for computing the M-orthogonal factorization (Figure 9), the vectors w_j are the auxiliary vectors that represent the vectors Mq_j. The matrix $\hat{B}$ is the triangular matrix computed in one iteration of the algorithm; the M-orthogonal triangular factor B is the product of all of the individual triangular matrices $\hat{B}$.

It should be noted that this algorithm may encounter a rank deficient set of vectors q_j and identically zero vectors are possible. Further details can be found in our discussion on when to terminate a Lanczos run (§4.4).

We have assumed in the discussion above that M is positive definite. In the case of M positive semidefinite, the recurrence, when properly started, generates a sequence of blocks, all of whose columns lie in the range of $(K - \sigma M)^{-1}M$. This is the subspace from which the eigenvectors corresponding to finite eigenvalues must be drawn [13]. Clearly the orthogonalization algorithm preserves this subspace. Further this subspace has only the trivial intersection with the nullspace of M [13, 28]. Thus, the appearance of a non-trivial column with zero M-norm represents a breakdown equivalent to rank deficiency, since such a vector cannot lie in the range of $(K - \sigma M)^{-1}M$.

4.2. Analysis of the Block Tridiagonal Matrix T_j. The original eigenvalue problem is reduced by the block Lanczos algorithm to an eigenvalue problem of the form

$$T_j s = s\theta,$$

where T_j is a block tridiagonal matrix, or equivalently, a symmetric $jp \times jp$ band matrix of bandwidth $2p+1$. Approximate eigenvectors for the original problem are found through the back transformation

$$y = Q_j s.$$

FIG. 10. *M-orthogonal Modified Gram-Schmidt Orthogonalization*

Initialization:

$Q = R$
$B = I$

Factorization:

Repeat
W = MQ

For $i = 1, 2, \ldots p$ do
$\hat{b}_{ii} = q_i^T w_i$
$q_i = q_i / \hat{b}_{ii}$
$w_i = w_i / \hat{b}_{ii}$
For $j = i + 1, \ldots p$ do
$\hat{b}_{ij} = q_i^T w_j$
$q_j = q_j - \hat{b}_{ij} q_i$
$w_j = w_j - \hat{b}_{ij} w_i$
End
End

$B = \hat{B} B$

Until convergence or iteration limit exceeded

In §2.2 we noted the standard result by which bounds on the accuracy of the computed eigenvalues can be computed without explicit computation of the eigenvectors. These bounds are used to determine whether to terminate the Lanczos recurrence and to evaluate which eigenpairs are accurate enough to be considered to have converged. The results in §2.2 generalize to provide a bound on the accuracy of the approximate eigenvalues of the spectrally transformed problem. However, our real interest is in the accuracy of our approximations to the original, untransformed, problem. We need to determine which eigenpairs of the original problem have converged, and we need accuracy estimates for all of the Ritz values for use in the shift selection process. To get these estimates we need to unravel the effects of the spectral transformation. Throughout we must account for possibly multiple eigenvalues.

Recall that the following relation (14) holds for vibration analysis:

$$(K - \sigma M)^{-1} M y - y\theta = Q_{j+1} B_{j+1} E_j^T s$$

Therefore, because Q_{j+1} is M-orthogonal,

$$\begin{aligned} \|M(K - \sigma M)^{-1} M - My\theta\|_{M^{-1}} &= \|MQ_{j+1} B_{j+1} E_j^T s\|_{M^{-1}} \\ &= \|B_{j+1} E_j^T s\|_2 \equiv \beta_j. \end{aligned}$$

Note that for each eigenvector s the corresponding β_j is the Euclidean norm of the product of the upper triangular matrix B_{j+1} with the last p components of s. We apply a theorem on the error in eigenvalue approximations for the generalized eigenproblem from [31] (pg. 318) to obtain:

$$|\frac{1}{\lambda - \sigma} - \theta| \leq \frac{\|M(K - \sigma M)^{-1} My - My\theta\|_{M^{-1}}}{\|My\|_{M^{-1}}} = \beta_j. \tag{18}$$

Thus, as in the ordinary eigenproblem, β_j is a bound on how well the eigenvalue of T_j approximates an eigenvalue of the operator to which the Lanczos algorithm is applied. We now extend this result to a bound on the error $|\lambda - \nu|$.

As in Ericsson and Ruhe's analysis [14], we use $|\frac{1}{\lambda - \sigma} - \theta| \leq \beta_j$ to show

$$\begin{aligned} |\lambda - \nu| &= |\lambda - \sigma - \frac{1}{\theta}| \\ &= |\frac{1}{\theta}(\lambda - \sigma)(\frac{1}{\lambda - \sigma} - \theta)| \\ &\leq \frac{1}{|\theta|}|\lambda - \sigma|\beta_j \leq \frac{\beta_j}{\theta^2}. \end{aligned}$$

The final inequality follows from the observation that the approximate values θ are derived from a projection onto a subspace; thus θ is always smaller than the eigenvalue $\frac{1}{\lambda - \sigma}$ it approximates. Hence

$$|\lambda - \nu| \leq \frac{\beta_j}{\theta^2}. \tag{19}$$

This shows how the accuracy requirements are modified by the spectral transformation. For an eigenvalue λ close to the shift σ we need only a moderately small β_j to guarantee a good

approximate eigenvalue ν because θ is large. Conversely, eigenvalues far from the shift are transformed to small values of θ, requiring smaller values of β_j than would otherwise be expected.

The bound (19) can be improved for well separated eigenvalues. Define the gap γ as:

$$\gamma \equiv \min_{\lambda_i \neq \lambda} |\frac{1}{\lambda_i - \sigma} - \frac{1}{\lambda - \sigma}|,$$

The gap bound theorem from [31](pg. 222) then results in

$$|\lambda - \nu| \leq \frac{\beta_j^2}{\theta^2 \gamma}. \tag{20}$$

Both bounds (19) and (20) are valid. In general the first is smaller than the second for clustered eigenvalues and larger for well separated eigenvalues. In our implementation we use whichever bound is smaller:

$$|\lambda - \nu| \leq \min\{\frac{\beta_j}{\theta^2}, \frac{\beta_j^2}{\theta^2 \gamma}\} \tag{21}$$

The definition of γ should be modified to account for clusters of eigenvalues; the gap between sets of multiple eigenvalues is used. In practice we have only an approximation to γ, which we derive from the shifted and inverted eigenvalues of T_j.

Similar error bounds can be derived for buckling analysis. Let (ν, y) be a computed eigenpair of (K, K_δ). Then

$$\theta = \frac{\nu}{\nu - \sigma}$$

and

$$|\frac{\lambda}{\lambda - \sigma} - \theta| \leq \beta_j.$$

From the fact that

$$\nu = \frac{\sigma\theta}{\theta - 1},$$

it follows that

$$\begin{aligned} \lambda - \nu &= \lambda - \frac{\sigma\theta}{\theta - 1} \\ &= (\frac{1}{\theta - 1})(\lambda(\theta - 1) - \sigma\theta) \\ &= (\frac{1}{\theta - 1})(\lambda - \sigma)(\theta - \frac{\lambda}{\lambda - \sigma}) \\ &= (\frac{\sigma}{\theta - 1})(\frac{\lambda - \sigma}{\sigma})(\theta - \frac{\lambda}{\lambda - \sigma}). \end{aligned}$$

Again, we use the property of the Lanczos algorithm of approximating eigenvalues from the inside. When the inversion of the operator is taken into account, the computed eigenvalues of the transformed problem are always closer to one than the true eigenvalues of the transformed problem. Therefore,

$$|\frac{\lambda-\sigma}{\sigma}| = \frac{1}{|\mu-1|} \leq \frac{1}{|\theta-1|}.$$

The resulting simple error bound for buckling analyses is

$$|\lambda-\nu| \leq \frac{|\sigma|}{(\theta-1)^2}\beta_j.$$

The analogous refined gap error bound is

$$|\lambda-\nu| \leq \frac{|\sigma|\beta_j^2}{(\theta-1)^2}\gamma_b,$$

where γ_b is defined by

$$\gamma_b \equiv \min_{\lambda_i \neq \lambda} |\frac{\lambda}{\lambda_i-\sigma} - \frac{\lambda}{\lambda-\sigma}|,$$

As in the vibration case, the lesser of the two bounds

$$|\lambda-\nu| \leq \min\{\frac{|\sigma|}{(\theta-1)^2}\beta_j, \frac{|\sigma|\beta_j^2}{(\theta-1)^2}\gamma_b\} \tag{22}$$

is chosen, with the definition of γ_b modified in the presence of multiple eigenvalues.

The spectral transformation preserves the eigenvectors, so there is no need to account for the transformation vis a vis the approximate eigenvectors. However, Ericsson and Ruhe introduced a correction term [14] that results in improved eigenvector approximations for the untransformed problem. This was later discovered to have the additional benefit [28] of ensuring that the computed eigenvectors lie in the proper subspace in cases where the metric matrix is semidefinite.

Let $\nu = \sigma + \frac{1}{\theta}$ be the computed eigenvalue. The correction step is formally one step of inverse iteration with the computed eigenvector y

$$(K-\sigma M)\tilde{z} = My$$

By (14)

$$\begin{aligned} \tilde{z} &= (K-\sigma M)^{-1}My \\ &= y\theta + Q_{j+1}B_{j+1}E_j^T s. \end{aligned}$$

The vector

$$z = \frac{1}{\theta}\tilde{z} = y + \frac{1}{\theta}Q_{j+1}B_{j+1}E_j^T s,$$

can be obtained cheaply by adding a linear combination of the next block of Lanczos vectors to y. This gives a better approximation to the eigenvector of the vibration problem, as well as ensuring that the approximate eigenvectors are uncontaminated by the effects of a semidefinite M. The corresponding correction for a semidefinite K in buckling analysis is given by

$$z = y + \frac{1}{\theta - 1} Q_{j+1} B_{j+1} E_j^T s.$$

During the course of the Lanczos algorithm we need to determine whether to continue or terminate the run. Key to this decision is knowledge of the convergence of the desired eigenvalues, as estimated by the residual bounds. To evaluate the bounds in (21) or (22) we need most of the eigenvalues and the corresponding entries in the bottom block row of the matrix of eigenvectors. Parlett and Nour-Omid [32] have a very efficient algorithm for the single vector Lanczos algorithm. Block generalizations have yet to be found, so we use a more straight forward approach. The eigenvalue problem for T_j is solved by reducing the band matrix T_j to tridiagonal form, and then by applying the tridiagonal QL algorithm. We use subroutines from EISPACK [16, 40], with slight modifications to obtain only the bottom p entries of the eigenvectors of T_j. These modifications reduce considerably both computation and storage requirements for each Lanczos step. Only $p^2 j$ words are needed as opposed to $(pj)^2$ for the full eigenvector matrix. We use the corresponding unmodified routines to obtain the full eigenvectors at the conclusion of a Lanczos run, at which time temporary space used during the recurrence is available to store the entirety of the eigenvector matrix for T.

4.3. Global Loss of Orthogonality and Reorthogonalization. Up to this point our discussion of the block Lanczos algorithm has assumed exact arithmetic, but the various error bounds hold in finite precision as well. It is well-known that the Lanczos algorithm misbehaves in inexact arithmetic. The most notable characteristic is the global loss of orthogonality among the computed Lanczos vectors. We expect $\mathcal{Q}_j$ to be an M-orthogonal matrix. In reality, finite precision arithmetic and no reorthogonalization spell disaster for this assumption. A reasonable correction is to perform *limited* reorthogonalization to keep $\mathcal{Q}_j$ sufficiently close to orthogonal. Our approach is two-fold — we identify mechanisms whereby orthogonality is lost and then apply a model of the loss of orthogonality to determine when to correct the situation. In the context of the block shifted Lanczos recurrence, orthogonality is lost in three different ways. First, there is a loss of orthogonality between adjacent blocks in $\mathcal{Q}_j$, the blocks the recurrence should make orthogonal. This is corrected by use of *local reorthogonalization.* Second, the blocks of $\mathcal{Q}_j$ not explicitly orthogonalized in the recurrence suffer a global loss of orthogonality. We correct for this with a block version of *partial reorthogonalization.* Lastly, it is important that a Lanczos run at some shift not recompute eigenvectors computed as a result of a previous Lanczos run. We present a new reorthogonalization scheme, *external selective reorthogonalization*, to ensure that this does not occur. Throughout the process our goal is to apply a minimal amount of extra work, particularly as it requires accessing the entirety of $\mathcal{Q}_j$, to maintain at least $\mathcal{O}(\sqrt{\epsilon})$-orthogonality in $\mathcal{Q}_j$.

The fundamental approach is to model the Lanczos recurrence in finite precision. The

following recurrence is our model of what really happens:

$$Q_{j+1}B_{j+1} = (K - \sigma M)^{-1}MQ_j - Q_jA_j - Q_{j-1}B_j^T + F_j. \tag{23}$$

In this model F_j represents the roundoff error introduced at step j. Then,

$$\begin{aligned} Q_k^T M Q_{j+1} B_{j+1} &= Q_k^T M (K - \sigma M)^{-1} M Q_j - Q_k^T M Q_j A_j \\ &\quad - Q_k^T M Q_{j-1} B_j^T + Q_k^T M F_j. \end{aligned}$$

Were $\mathcal{Q}_j$ truly M-orthogonal, all the quantities $Q_k^T M Q_l, k \neq l$ would be zero. These quantities cannot be ignored in the presence of roundoff error. For convenience we define

$$W_{j,k} \equiv Q_k^T M Q_j,$$

with which the previous equation becomes

$$W_{j+1,k}B_{j+1} = Q_k^T M(K - \sigma M)^{-1}MQ_j - W_{j,k}A_j - W_{j-1,k}B_j^T + Q_k^T MF_j. \tag{24}$$

This equation is nearly sufficient for our computational purposes. We can easily find norms for the blocks A_j and B_j during the recurrence, and we will compute bounds for all the other terms except for the first term on the right side of (24). We eliminate $Q_k^T M(K - \sigma M)^{-1}MQ_j$ from (24) by obtaining an expression for its transpose by premultiplying the occurrence of (23) with $j = k$ by $Q_j^T M$:

$$Q_j^T M(K - \sigma M)^{-1}MQ_k = W_{j,k+1}B_{k+1} + W_{k,j}A_j + W_{k-1,j}B_k^T + Q_j^T MF_k.$$

The obvious substitution then results in

$$\begin{aligned} W_{j+1,k}B_{j+1} &= B_{k+1}^T W_{j,k+1} + A_k W_{j,k} + B_k W_{j,k-1} \\ &\quad - W_{j,k}A_j - W_{j-1,k}B_j^T + G_{j,k}. \end{aligned} \tag{25}$$

Here $G_{j,k} \equiv Q_k^T MF_j - F_k^T MQ_j$ represents the local roundoff error. Formula (25) explains the global loss of orthogonality. We will use this model to estimate and bound the loss of orthogonality among the Lanczos vectors, and thereby determine how to correct the loss of orthogonality.

4.3.1. Monitoring the Loss of Orthogonality. The development of our modeling procedure has two parts, both based on the bounds available by taking norms of (25):

$$\begin{aligned} \|W_{j+1,k}\|_2 \leq\ & \|B_{j+1}^{-1}\|_2(\|B_{k+1}\|_2\|W_{j,k+1}\|_2 \\ & + \|B_k\|_2\|W_{j,k-1}\|_2 + \|B_j\|_2\|W_{j-1,k}\|_2 \\ & + (\|A_j\|_2 + \|A_k\|_2)\|W_{j,k}\|_2 + \|G_{j,k}\|_2). \end{aligned}$$

We use this equation to compute a bound $\omega_{j,k}$ on $\|W_{j,k}\|_2$ at each step.

The first part of our development addresses the bounds $\omega_{j+1,k}$ for $k \leq j - 1$, that is, for blocks that are not explicitly involved in the orthogonalization of the Lanczos vectors within the recurrence itself. For these blocks the loss of orthogonalization depends on the

FIG. 11. *Simulation of Loss of Orthogonality (ω-recurrence)*

Initialize:

$\epsilon_s \equiv \epsilon p\sqrt{n}$, where $\epsilon \equiv$roundoff unit, p is the block size
and n = number of degrees of freedom

$\omega_{2,1} = \epsilon_s$

Loop:

For $j = 3, 4, \ldots$ do
$\omega_{j+1,j} = \epsilon_s$
$\omega_{j+1,j-1} = \tilde{\beta}_{j+1}(2\beta_j\epsilon_s + (\alpha_j + \alpha_{j-1})\epsilon_s + \beta_{j-1}\omega_{j,j-2})$
For $k = 1, \ldots j-2$ do
$\omega_{j+1,k} = \tilde{\beta}_{j+1}(\beta_{k+1}\omega_{j,k+1} + \beta_k\omega_{j,k-1} + \beta_j\omega_{j-1,k} + (\alpha_j + \alpha_k)\omega_{j,k})$
End
End

loss already incurred at previous steps. Bounds on that loss of orthogonality will be available to us from previous steps of the simulation given in Figure 10.

The following quantities from the Lanczos recurrence are required for the simulation:

$$\begin{aligned}
\alpha_k &\equiv \|A_k\|_2 \\
\beta_k &\equiv \|B_k\|_2 \\
\tilde{\beta}_k &\equiv 1/\sigma_p(B_k), \text{ where } \sigma_p(B_k) \text{ is the smallest singular value of } B_k.
\end{aligned}$$

In addition, we follow [31, 36, 38] in making a standard assumption on a bound for the error term $\|G_{j,k}\|_2 \leq \epsilon p\sqrt{n}$. We have as yet left unstated the origin of the two initializing terms $\omega_{j+1,j}$ and $\omega_{j+1,j-1}$. In examining them we will uncover a particular artifact of the *block* Lanczos algorithm. By (25),

$$\begin{aligned}
W_{j+1,j-1}B_{j+1} &= B_j^T W_{j,j} + A_{j-1}W_{j,j-1} + B_{j-1}W_{j,j-2} \\
&\quad -W_{j,j-1}A_j - W_{j-1,j-1}B_j^T + G_{j,j-1} \\
&= (B_j^T W_{j,j} - W_{j-1,j-1}B_j^T) \\
&\quad +(A_{j-1}W_{j,j-1} - W_{j,j-1}A_j) \\
&\quad +B_{j-1}W_{j,j-2} + G_{j,j-1}.
\end{aligned}$$

By reason of the care with which we compute the QR factorization, we assume that $Q_j^T M Q_j = I + E$, where I is the identity matrix and $\|E\|_2 \leq \epsilon_s$. For reasons discussed below, we can assume that $\|W_{j,j-1}\|_2 \leq \epsilon_s$. From this it follows that

$$\|W_{j+1,j-1}\|_2 \leq \tilde{\beta}_{j+1}(2\beta_j\epsilon_s + (\alpha_j + \alpha_{j-1})\epsilon_s + \beta_{j-1}\omega_{j,j-2}). \tag{26}$$

Notice from (26) that $\omega_{j+1,j-1} > \tilde{\beta}_{j+1}\beta_j\epsilon_s$. At the next step this term will appear as $\tilde{\beta}_{j+2}\beta_j\omega_{j+1,j-1}$; in the following step it will be one of the contributions to $\tilde{\beta}_{j+3}\beta_{j+1}\omega_{j+2,j}$.

Both $\tilde{\beta}_{j+1}$ and β_{j+1} appear in this last product. The growth of the bound occurs as fast as $\kappa(B_j)$, where $\kappa(B_j) = \tilde{\beta}_{j+1}\beta_{j+1}$ is the condition number of B_j. The analysis of the ordinary Lanczos algorithm has unity corresponding to the term $\kappa(B_j)$, because the condition number of a non-zero 1×1 matrix is always one. The loss of orthogonality occurs more rapidly in the block Lanczos algorithm, particularly when $\kappa(B_j)$ is significantly larger than one, but also in general.

A different, but related, analysis can be used to show that the term $\kappa(B_j)$ appears in the bound for $\omega_{j+1,j}$. This was first observed in [22], where this growth was also actually observed in the Lanczos recurrence. An inexpensive correction is needed to make the recurrence useful: at each step a *local reorthogonalization* between Q_{j+1} and Q_j is performed. Because the Lanczos recurrence is itself just a special form of Gram-Schmidt orthogonalization, local reorthogonalization can be seen as a simple generalization of the reorthogonalization required in computing the M-orthogonal factorization of a single block. Local reorthogonalization ensures that ϵ_s orthogonality holds between successive blocks of Lanczos vectors. Note that a local orthogonalization step is also performed on completion of a partial reorthogonalization. If storage is not an issue, a local reorthogonalization between Q_{j+1} and Q_{j-1} should also be performed, in which the obvious modification should be made to the algorithm for computing the ω-recurrence.

4.3.2. Partial Reorthogonalization. The global loss of orthogonality modeled by the ω-recurrence can be corrected by two different schemes. These are the selective orthogonalization scheme of Parlett and Scott [34] and the partial reorthogonalization scheme of Simon [39]. Selective orthogonalization is an elegant scheme that takes advantage of the special characteristics of the loss of orthogonality — orthogonality is lost exactly in the directions of eigenvectors that have become well-represented in $\mathcal{Q}_j$. Selective orthogonalization is implemented in two steps. In the first, the Lanczos recurrence is "interrupted" when an eigenvector converges. The eigenvector is computed, which requires access to all previous blocks in $\mathcal{Q}_j$. The second step occurs whenever the model indicates orthogonality is lost again in the direction of the eigenvector. The second step requires that the latest two Lanczos blocks be reorthogonalized against the computed eigenvector, but does not require access to preceding blocks of $\mathcal{Q}_j$.

Partial reorthogonalization is a simpler, but less elegant, scheme. Whenever the simulation indicates too great a loss of orthogonality, interrupt the recurrence to reorthogonalize Q_j and Q_{j+1} against all preceding blocks. Each step at which reorthogonalization occurs represents access to all of $\mathcal{Q}_j$. For this reason partial reorthogonalization has previously been recommended for situations in which the eigenvectors were not of any interest (as in solving sparse linear equations [39]). The extra cost in an application of partial reorthogonalization does have an extra payoff; orthogonality is restored against all converged and nearly converged eigenvectors simultaneously.

Shifting and the block recurrence each accelerate the convergence of eigenpairs. The simultaneous use of both means that eigenpairs converge very rapidly. Frequently one or more eigenpairs converge at *each* block step, once the recurrence is established. In this circumstance selective orthogonalization has possibly greater requirements for accessing $\mathcal{Q}_j$ than does partial reorthogonalization. Selective orthogonalization will require an eigenvector

computation at almost each step; partial reorthogonalization will occur only every three to four steps in typical problems. It would be possible to combine the two schemes — to carry out partial reorthogonalization during the computation of an eigenvector for selective orthogonalization, but it is not clear that the combination would be more effective than partial reorthogonalization alone. (See [33] for a discussion of these issues for the ordinary Lanczos recurrence.) Table 6 summarizes the reorthogonalization requirements of two extensive eigencomputations. The number of selective orthogonalization steps given in this table is the number of block steps at which one or more eigenvalues converge; the number of partial reorthogonalization steps is the number of block steps at which partial reorthogonalization was performed.

Our implementation of the Lanczos recurrence is based, therefore, on the block generalization of partial reorthogonalization, based on the block ω-recurrence presented above. The single vector version of this simulation has been shown previously [39] to provide a good order of magnitude estimate of growth of the loss of orthogonality as well as a bound. We use the block version to estimate the loss of orthogonality to determine when reorthogonalization is necessary. Previous work [31, 34, 38] indicates that reorthogonalization is needed whenever

$$\max_k \omega_{j+1,k} \geq \sqrt{\epsilon}.$$

The reorthogonalization should be carried out with both of the last two block of vectors Q_j and Q_{j+1}, in order that the next block generated by the recurrence, Q_{j+2}, be strictly orthogonal to all of its predecessors. This leads to the following *partial reorthogonalization* [39] algorithm (Figure 11) for maintaining orthogonality:

FIG. 12. *Partial Reorthogonalization*

At each Lanczos step, after computing Q_{j+1} and B_{j+1} do:

Update the ω-recurrence as above

$\omega_{max} \equiv \max_k \omega_{j+1,k}$
If $\omega_{max} \geq \sqrt{\epsilon}$ then
  For $k = 1, \ldots j-1$ do
    Orthogonalize Q_j against Q_k
    Orthogonalize Q_{j+1} against Q_k
  End
  Orthogonalize Q_{j+1} against Q_j
  Reinitialize ω-recurrence:
    $\omega_{j+1,k} = \omega_{j,k} = \epsilon_s, k = 1, \ldots j$
End if

Note that the orthogonalization of Q_j and Q_{j+1} involves M-inner products. This requires the storage of both the Lanczos vectors and their product with M in secondary storage, or, alternatively, reapplying M to the Lanczos vectors. The appropriate form depends on cost.

TABLE 5
Comparison of Partial and Selective Reorthogonalization

matrix	shift	eigen-values	block steps	partial reorthog. steps	selective orthog. steps
BCSST_26[a]	1	60	28	9	20
	2	15	20	5	10
	3	37	21	9	15
	4	31	13	3	8
	5	10	10	2	5
	6	26	17	5	13
	7	22	16	5	11
	8	6	3	1	3
	total	207	159	41	83
PLAT1919[b]	1	8	17	5	5
	2	83	31	8	20
	3	29	17	6	9
	4	24	21	5	10
	5	39	21	6	14
	6	55	31	12	22
	7	8	11	3	4
	8	27	13	3	8
	9	32	25	7	16
	10	1	7	1	1
	11	8	11	4	4
	12	62	35	10	28
	13	24	18	6	8
	14	36	18	8	12
	15	20	16	4	8
	16	74	36	14	29
	total	636	404	102	200

[a] block size 3, lowest 200 modes
[b] block size 5, all modes in [.000025, .24]

4.3.3. External Selective Orthogonalization. A different type of loss of orthogonality occurs in the context of the shifted and inverted Lanczos algorithm. It is possible that, after computing some eigenvalues with shift σ_1, the same eigenvalues and vectors are computed again when using σ_2. This presents a severe complication — we need a mechanism for identifying duplicate copies from different runs. In addition, we waste resources recomputing eigenvectors. In order to avoid this complication and the duplicate computation we have developed another reorthogonalization scheme, *external selective orthogonalization.* External selective orthogonalization is an efficient way of keeping the current sequence of Lanczos vectors orthogonal to previously computed eigenvectors, and thereby avoiding the recomputation of eigenvalues that are already known. External selective orthogonalization is motivated by the classical selective orthogonalization algorithm [34], but the development here is entirely new.

In theory it would be sufficient to orthogonalize the starting block against known eigenvectors, because this would guarantee that all subsequent Lanczos vectors are orthogonal as well. Of course this does not hold in practice. A global loss of orthogonality occurs, similar to the one among the Lanczos vectors themselves; in addition, the computed eigenvector is not exact. The contribution of both sources of error, which ultimately may lead to the recomputation of eigenvalues and vectors, is analyzed below.

Let (ν, y) be an approximate eigenpair of (K, M). For clarity, denote the current shift as σ_{new}. The relationship between the eigenvector y and the Lanczos vectors obtained with the shift σ_{new} is found by premultiplying the finite precision recurrence (23) by $y^T M$ to obtain

$$\begin{aligned} y^T M Q_{j+1} B_{j+1} \quad &= y^T M (K - \sigma_{new} M)^{-1} M Q_j - y^T M Q_j A_j \\ &\quad - y^T M Q_{j-1} B_j^T + y^T M F_j, \end{aligned} \tag{27}$$

which includes the effect of the local error term.

We assume that B_{j+1} is nonsingular. Then we can obtain a bound on the loss of orthogonality between y and Q_j by taking norms of both sides of (27):

$$\begin{aligned} \|y^T M Q_{j+1}\|_2 \quad &\le \|B_{j+1}^{-1}\|_2 (\|y^T M (K - \sigma_{new} M)^{-1} M Q_j\|_2 + \|y^T M Q_j\|_2 \|A_j\|_2 \\ &\quad + \|y^T M Q_{j-1}\|_2 \|B_j^T\|_2 + \|y^T M F_j\|_2). \end{aligned}$$

As with partial reorthogonalization, we can define a recurrence relation for a quantity τ_j to bound the loss of orthogonality between y and the Lanczos vectors. Assuming that $\tau_i \ge \|y^T M Q_i\|_2$ for $i = 1, \ldots, j$, we obtain

$$\|B_{j+1}^{-1}\|_2 (\|y^T M (K - \sigma_{new} M)^{-1} M Q_j\|_2 + \tau_j \|A_j\|_2 + \tau_{j-1} \|B_j^T\|_2 + \|y^T M F_j\|_2).$$

as a bound for the right-hand side of the $j+1$-st step. Of the four terms on the right hand side of this equation, the second and third are easily computed. For the fourth we shall use the same assumption as before on the size of the local error term: $\|F_j\|_2 \le \epsilon p \sqrt{n}$. We need then only to bound the first term $\|y^T M (K - \sigma_{new} M)^{-1} M Q_j\|_2$. The spectral transformations preserve eigenvectors, so y is also an approximate eigenvector of the spectrally transformed problem. Define the transformed residual vector z_{new} by

$$(K - \sigma_{new} M)^{-1} M y - \frac{1}{\nu - \sigma_{new}} y = z_{new}.$$

Then

$$y^T M(K - \sigma_{new} M)^{-1} M Q_j = \frac{1}{\nu - \sigma_{new}} y^T M Q_j + z_{new}^T M Q_j,$$

from which it follows that

$$\|y^T M(K - \sigma_{new} M)^{-1} M Q_j - y^T M Q_j A_j\|_2 \leq \|(\frac{1}{\nu - \sigma_{new}} I - A_j)\|_2 \tau_j + \|z_{new}^T M Q_j\|_2.$$

But

$$\begin{aligned} \|z_{new}^T M Q_j\|_2 &= \|(z_{new}^T M^{1/2})(M^{1/2} Q_j)\|_2 \\ &\leq \|z_{new}^T M^{1/2}\|_2 \|M^{1/2} Q_j\|_2 = \|z_{new}^T\|_M \|Q_j\|_M = \|z_{new}^T\|_M \end{aligned}$$

Thus, the following simple recurrence for τ gives a bound for the loss of orthogonality observed in (27):

$$\tau_{j+1} = \|B_{j+1}^{-1}\|_2 (\tau_j \|(\tfrac{1}{\nu - \sigma_{new}} I - A_j)\|_2 + \tau_{j-1} \|B_j^T\|_2 + \|z_{new}\|_M + \epsilon p \sqrt{n}). \tag{28}$$

The same analysis applies to the buckling spectral transformation, where the eigenvector orthogonality error equation (27) becomes:

$$\begin{aligned} y^T K Q_{j+1} B_{j+1} &= y^T K (K - \sigma_{new} K_\delta)^{-1} K Q_j - y^T K Q_j A_j \\ &\quad - y^T K Q_{j-1} B_j^T + y^T K F_j. \end{aligned}$$

The transformed residual vector z_{new} is

$$(K - \sigma_{new} K_\delta)^{-1} K y - \frac{\nu}{\nu - \sigma_{new}} y = z_{new}.$$

By the same analysis as above, the recurrence for τ in the buckling context is

$$\tau_{j+1} = \|B_{j+1}^{-1}\|_2 (\tau_j \|(\tfrac{\nu}{\nu - \sigma_{new}} I - A_j)\|_2 + \tau_{j-1} \|B_j^T\|_2 + \|z_{new}\|_K + \epsilon p \sqrt{n}). \tag{29}$$

The recurrences in (28) and (29) provide a mechanism for estimating the loss of orthogonality to externally computed eigenvectors, regardless of the source. Each requires computing the transformed residual vector, z_{new}, and its norm, but the recurrence applies to situations where eigenvectors are known adventitiously. For example, in the vibration analysis of structures where K is singular, the so-called rigid body modes, the zero eigenvalues and vectors, often can be computed at much less cost than a factorization. Typically, the cost of computing the residual norms for all of the vectors involved in external selective orthogonalization is less than the cost of one additional step of the Lanczos recurrence.

In the context of a Lanczos code within a larger shifting strategy, it would be attractive to use the information from the Lanczos recurrence to bound the errors in the computed eigenvectors and thereby avoid having to compute $\|z_{new}\|_M$. The following analysis applies when the approximate eigenpair (ν, y) was computed by the Lanczos code at a previous shift σ_{old}.

The approximate eigenpair (ν, y) was computed with shift σ_{old} by applying the Ericsson-Ruhe correction to the pair (ν, w). Define the residual vector z_{old} by

$$(K - \sigma_{old}M)^{-1}My - \theta_{old}y = z_{old},$$

where

$$\theta_{old} = \frac{1}{\nu - \sigma_{old}}.$$

By the Ericsson-Ruhe correction, using Q_{j+1} and B_{j+1} from the run at σ_{old},

$$y = w + \frac{1}{\theta_{old}}Q_{j+1}B_{j+1}E_j^T s.$$

Then

$$\begin{aligned}(K - \sigma_{old}M)^{-1}My - \theta_{old}y = \\ [(K - \sigma_{old}M)^{-1}Mw - \theta_{old}w - Q_{j+1}B_{j+1}E_j^T s] + \\ \frac{1}{\theta_{old}}(K - \sigma_{old}M)^{-1}MQ_{j+1}B_{j+1}E_j^T s.\end{aligned}$$

But the terms inside square brackets on the right hand side constitute the residual equation for w and sum to zero; hence,

$$(K - \sigma_{old}M)^{-1}My - \theta_{old}y = \frac{1}{\theta_{old}}(K - \sigma_{old}M)^{-1}MQ_{j+1}B_{j+1}E_j^T s.$$

If we apply the obvious bound to the right hand side, we find

$$(30) \qquad \|z_{old}\|_M = \|(K - \sigma_{old}M)^{-1}My - \theta_{old}y\|_M \leq |\frac{1}{\theta_{old}}|\|(K - \sigma_{old}M)^{-1}M\|_M\beta_j.$$

Here $\|Q_{j+1}B_{j+1}E_j^T s\|_M \leq \beta_j$ is the standard residual bound (18), and we note that β_j is small because we accepted the eigenpair (ν, y).

For symmetry, let $\theta_{new} = \frac{1}{\nu - \sigma_{new}}$. A simple relationship between the old shift σ_{old} and residual z_{old}, and the new shift σ_{new} and residual z_{new} can be obtained from the *untransformed* residual vector. Define d by

$$Ky - \nu My = d.$$

For *any* $\sigma \neq \nu$

$$(K - \sigma M)y + (\sigma - \nu)My = d,$$

or

$$My = \frac{1}{\nu - \sigma}((K - \sigma M)y - d)$$

Therefore, for any choice of $\sigma \neq \nu$,

$$\begin{aligned} (K-\sigma M)^{-1}My &= \frac{1}{\nu-\sigma}(y-(K-\sigma M)^{-1}d) \\ &\text{or} \\ (K-\sigma M)^{-1}My - \frac{1}{\nu-\sigma}y &= -\frac{1}{\nu-\sigma}(K-\sigma M)^{-1}d \end{aligned}$$

For the particular shifts σ_{old} and σ_{new} we obtain

$$\begin{aligned} z_{old} &= -\frac{1}{\nu-\sigma_{old}}(K-\sigma_{old}M)^{-1}d \\ z_{new} &= -\frac{1}{\nu-\sigma_{new}}(K-\sigma_{new}M)^{-1}d, \end{aligned}$$

from which it follows that

$$z_{new} = \frac{\nu-\sigma_{old}}{\nu-\sigma_{new}}(K-\sigma_{new}M)^{-1}(K-\sigma_{old}M)z_{old},$$

This result, together with the definition of z_{new}, produces

$$\begin{aligned} (K-\sigma_{new}M)^{-1}My-\theta_{new}y &= \frac{\nu-\sigma_{old}}{\nu-\sigma_{new}}(K-\sigma_{new}M)^{-1}(K-\sigma_{old}M)z_{old} \\ &= \frac{\nu-\sigma_{old}}{\nu-\sigma_{new}}(K-\sigma_{new}M)^{-1} \\ &\quad ((K-\sigma_{new}M)+(\sigma_{new}-\sigma_{old})M)z_{old} \\ &= \frac{\nu-\sigma_{old}}{\nu-\sigma_{new}}(I+(\sigma_{new}-\sigma_{old})(K-\sigma_{new}M)^{-1}M)z_{old}. \end{aligned}$$

We can bound the norm of this term by

$$\begin{aligned} &\|z_{new}\|_M \leq \\ &\frac{|\nu-\sigma_{old}|}{|\nu-\sigma_{new}|}(1+|\sigma_{new}-\sigma_{old}|\|(K-\sigma_{new}M)^{-1}M\|_M)\frac{\|(K-\sigma_{old}M)^{-1}M\|_M}{\theta_{old}}\beta_j. \end{aligned} \tag{31}$$

The two matrix norms in (31) are the norms of the operator matrices to which the Lanczos algorithm is applied at shifts σ_{new} and σ_{old}. Thus, each is estimated by the norms of the corresponding tridiagonal matrices. It is difficult to see how to use (31) with shift σ_{new} because the tridiagonal matrix is being created at the same time in which the bound is needed.

A context in which this bound may be more useful is when successive Lanczos runs are made with the same shift. This usually occurs when the Lanczos algorithm is applied with a block size much smaller than the actual multiplicities of eigenvalues. In such a case, $z_{new} = z_{old}$, and we can use (30) instead of (31):

$$\|z_{new}\|_M \leq \frac{\|(K-\sigma_{old}M)^{-1}M\|_M}{\theta_{old}}\beta_j.$$

Here, $\|T_{old}\|_2$ is available as an estimate of $\|(K-\sigma_{old}M)^{-1}M\|_M$. Normally one would expect this to be a reliable estimate, but it is not guaranteed to be so if the largest eigenvalues of $(K-\sigma_{old}M)^{-1}M$ are themselves suppressed by selective orthogonalization. The estimate of $\|(K-\sigma_{old}M)^{-1}M\|$ should therefore be taken as the maximum of the largest eigenvalue of all the (block) tridiagonal matrices produced with this shift, $\sigma_{new} = \sigma_{old}$, and of the spectral transformation of the eigenvalue nearest σ_{old} that is suppressed by external selective orthogonalization.

This analysis is predicated on the approximate eigenpairs (ν, y) of the original problem holding at least semi-orthogonality amongst themselves. If this assumption is false, it would be necessary to obtain, via a robust Gram-Schmidt process, an orthogonal basis for the space spanned by the vectors involved in external selective orthogonalization.

As with partial reorthogonalization, we define a recurrence relation for a quantity τ_j that estimates the loss of orthogonality of the Lanczos vectors with respect to y. In the recurrence τ_j is defined be:

$$\tau_{j+1} = \tilde{\beta}_{j+1}(\alpha_{\nu\sigma j}\tau_j + \beta_j\tau_{j-1} + \delta), \tag{32}$$

where we set initially $\tau_0 \equiv 0$ and $\tau_1 = \epsilon p\sqrt{n}$. The terms β_j and $\tilde{\beta}_{j+1}$ are defined as in the ω-recurrence. The term $\alpha_{\nu\sigma j} \equiv \|(\nu-\sigma)^{-1}I - A_j\|_2$. The final term δ is taken to be $\|z_{new}\|_M$ as explicitly computed or possibly as given by the bound above.

Whenever $\tau_{j+1} \geq \sqrt{\epsilon}$ an external selective orthogonalization is performed in order to ensure that the sequence of Lanczos vectors remains orthogonal to working precision to the computed eigenvectors. It should be noted that a relatively large residual for the computed eigenvector will cause frequent reorthogonalization, but as noted below, usually only a very small number of vectors are actually involved. External selective orthogonalization is implemented as in Figure 12

FIG. 13. *External Selective Orthogonalization*

Before the Lanczos iteration:
    Determine the set of SO-vectors (eigenvectors for selective orthogonalization)
    Orthogonalize Q_1 against the SO-vectors.
    Orthogonalize Q_2 against the SO-vectors.

At each Lanczos step $j = 3, 4, \ldots$ do:
    Update the τ-recurrence according to (32) for each SO-vector;
    If (τ_j has been greater than $\sqrt{\epsilon}$ or $\tau_{j+1} \geq \sqrt{\epsilon}$) then
        Orthogonalize Q_{j+1} against y
        Set $\tau_{j+1} = \epsilon_s$
    End if

It is unnecessary to perform external selective orthogonalization against *all* previously computed eigenvectors. From (28, 29) it is evident that one of the main driving forces in the loss of orthogonality is $(\nu-\sigma)^{-1}$. It would appear that loss of orthogonality should

mostly occur in the direction of eigenvectors corresponding to eigenvalues close to the new shift. Further, as discussed in §3.3, only a few eigenvectors, again usually those close to the new shift, need be considered in order to avoid confusing new eigenvectors with old. In our implementation, we use sentinels to reduce the cost of maintaining orthogonality. The set of eigenvectors used for external selective orthogonalization is usually the eigenvectors corresponding to any known eigenvalues closer to the shift than the sentinels . Eigenvalues beyond the sentinels are discarded in the analysis of the block tridiagonal system.

The effect of using sentinels on the work required for external selective orthogonalization is more dramatic than is suggested by the analysis above. Although proximity to the shift is the driving force in the growth of τ, neither recurrence (28, 29) begins at ϵ. The term $\|z_{new}\|_M$ is usually near $\sqrt{\epsilon}$. The eigenvalues themselves are only good to the convergence tolerance (usually $\epsilon^{2/3}$ in our code). Further, the spectral transformations preserve eigenvectors, but do not preserve the property of being the best minimizers for approximate eigenvalues (see [14] for a discussion of the need to modify the approximate eigenvectors). As a result, external selective orthogonalization happens more often than we might expect, often at every step for the eigenpairs nearest the sentinels, which often are the least accurate as well as the ones nearest the new shift.

Experimental results are shown for two examples in Table 7. The results shown as "with sentinels" refers to the selection described in §3.3; the results shown as "without sentinels" uses as SO-vectors all eigenvectors in the intervals between the current shift and any neighboring trust intervals. The figure given as "cpu cost" includes both cpu time and i/o processor time. The difference between the costs for the two variations gives only a rough idea of the added cost for complete selective orthogonalization because the difference in cost affects the termination decision for each run and thereby changes the choice of shifts.

TABLE 6
External Selective Orthogonalization

matrix	with sentinels			without sentinels		
	average number of S.O. Vectors	total number of S.O. Steps	cpu cost	average number of S.O. Vectors	total number of S.O. Steps	cpu cost
BCSST_26[a]	1.9	391	207	11.0	2106	243
PLAT1919[b]	3.8	1268	1044	28.4	6256	1099

[a] block size 3, lowest 200 modes
[b] block size 5, all modes in [.000025, .24]

The orthogonalizations involve again both y and My. In order to avoid the repeated computation of My, all selective orthogonalization vectors are premultiplied by M and the result is stored on the same random access file as the eigenvectors y. This computation is performed before the actual Lanczos run begins.

4.3.4. Summary of Reorthogonalization Schemes. We now present in summary form the reorthogonalized block Lanczos algorithm as used in our production code for solving

vibration problems. (Buckling analysis differs only in using K-orthogonality and in substituting K_δ for M.) Our scheme consists of applying, in turn, external selective, partial and local reorthogonalization to the result of a single block Lanczos step. The first two schemes are applied only when the respective model signals a need therefor; each should be applied before orthogonality is lost badly enough that repeated orthogonalization are needed. The local reorthogonalization is applied at each step. It may be applied repeatedly, but this normally occurs only when the recurrence has broken down, which will cause termination. The integration of these is indicated in Figure 13.

FIG. 14. *Block Lanczos Algorithm Preserving Semi-Orthogonality For the Vibration Problem*

Initialization:

Set $Q_0 = 0$
Set $B_1 = 0$
Choose R_1 and orthonormalize the columns of R_1 to obtain Q_1
with $Q_1^T(MQ_1) = I_p$.

Lanczos Loop:

For $j = 1, 2, 3 \ldots$ do

Set $U_j = (K - \sigma M)^{-1}(MQ_j) - Q_{j-1}B_j^T$
Set $A_j = U_j^T(MQ_j)$
Set $R_{j+1} = U_j - Q_j A_j$
Compute Q_{j+1} and (MQ_{j+1}) such that
a) $Q_{j+1}B_{j+1} = R_{j+1}$
b) $Q_{j+1}^T(MQ_{j+1}) = I_p$

Evaluate γ-recurrence for each SO vector and perform selective
orthogonalization if necessary
Evaluate ω-recurrence and perform partial reorthogonalization if necessary

Repeat up to p times:
Reorthogonalize Q_{j+1} to Q_j
Recompute M-orthogonal factor of Q_{j+1}
Until orthogonal factorization occurs in one step

End loop

4.4. Cost Analysis and Termination of a Lanczos Run. The block Lanczos algorithm exists as part of a larger code, in which each Lanczos run is intended to solve only a subproblem. In this environment we expect to make a number of Lanczos runs with different

shifts. There are three ways in which a given Lanczos run can terminate:

1. All eigenvalues required for this subproblem have converged.
2. The B_{j+1}-block is ill-conditioned or singular. In this case a continuation of the Lanczos run is either numerically difficult or impossible. Singular or ill-conditioned B_{j+1}-blocks can be encountered for the following reasons:
   - The shift is very close to an eigenvalue.
   - The effective space of Lanczos vectors is exhausted — we cannot compute more orthogonal vectors than the problem has finite eigenvalues.
   - Dependencies within the starting block cause a singular B_{j+1} at some later stage.
3. Eigenvalues farther from the shift appear to be converging slowly. The estimated cost for computing them in the current Lanczos run is great enough that a new shift should be chosen.

The first of these is easy to detect in most cases, assuming that the accuracy of the Ritz values can be evaluated (§4.2) and that the shift selection has communicated the number of eigenvalues desired. There is a minor complication in the case for eigenvalues desired closest to some specified value. We do not know in advance how many eigenvalues are required on each side of ξ. Our code is conservative — at a given shift it looks for as many eigenvalues as are required to complete the total, even if these may represent more than what is needed on a single side of ξ. As a result, we may not terminate as early as would be appropriate from hindsight.

Breakdown in the recurrence is perhaps more likely than might otherwise be expected. The first two reasons for breakdown occur in practice; we have never seen the third occur. The first we try to avoid during the shift selection process; the second occurs primarily during user evaluation of the code, when it is not uncommon to be faced with problems like finding all of the eigenvalues of 7×7 matrices using a blocksize of 5. In all three cases breakdown is detected by one of two mechanisms. Either the norm of the residual block is very small compared to the norm of the diagonal block or the off-diagonal block is ill-conditioned and presumed rank-deficient. We use a relative norm of $1/\sqrt{\epsilon}$ for the first case. For the second we compute, at each step, the extreme singular values of the off-diagonal block B_i; we terminate if the condition number of $B_i \geq 1/\epsilon$. Since we really want only the condition number of B_i, our use of singular values is perhaps overkill. However, the cost of a singular value decomposition of a $p \times p$ matrix is trivial compared to the cost of an $n \times n$ block solve. There are certainly other options open to us.

The most common reason for termination is that computing more eigenvalues in the current run is inefficient. This decision to stop is based on a cost analysis that is carried out at each Lanczos step. The cost analysis assumes that a measure of the real user cost is available, which is used to monitor the cost of the various operations in the algorithm. This is used in a model of the Lanczos algorithm, which estimates the total cost of continuing the Lanczos run beyond the current step. The residual bounds estimating the accuracy of yet unconverged eigenvalues are monitored step by step; the observed changes are used to estimate future convergence. We use our model to attempt to locate a point in an individual run where the average cost per eigenvalue is minimized. This in turn is a heuristic

attempt to minimize the average cost for all eigenvalues. The effectiveness of the heuristic is demonstrated later for a particular example in Table 14.

The cost of a Lanczos run depends on a number of parameters. There is usually a factorization associated with the run. The cost of the factorization is typically the largest single cost, but it is a one-time cost. Each Lanczos step has several components of cost. There is a large constant cost per step, comprising the matrix-block solve and multiplication and other operations in the recurrence. There are costs that increase quadratically in the number of block steps, notably the cost of the eigenanalysis of T_j. We also expect the cost of partial reorthogonalization to increase at least quadratically. Partial reorthogonalization occurs primarily as a function of the nearness of eigenvalues to the shift. Inasmuch as the eigenvalue nearest the shift is usually the first to converge, and dominates the reappearance of banished subspaces, the frequency with which partial reorthogonalization is needed is generally independent of the number of eigenvalues that have converged. There are also costs that increase cubically, particularly the cost of computing the converged eigenvectors from the Lanczos vectors, an operation that occurs only after the run terminates.

Normally, eigenvalues far from the shift converge slowly and require a large number of steps. We expect to see the fastest convergence early, with the number of eigenvalues converging per step tapering off as the length of the run increases. At first the cost *per eigenvalue* decreases rapidly, as the cost of the factorization is divided among the converged values. Later the other costs of the iteration come to dominate; as the convergence rate slows and the costs increase, the average cost also increases. Our goal is to stop at the minimum average cost.

The costs of the various operations vary dramatically from system to system and problem to problem. A particular concern for a commercial code is input/output cost, normally not an issue at all for academic analyses. Rather than adopting a simplified model of individual pieces of the cost, we have made the assumption that the real system cost is available. We use a cubic model of cost, matching the dependence of the various pieces on j. We obtain a least squares fit to the real cost over the last 10 steps and use this as a model to predict the real cost over the new few steps. This model is supplemented by an estimate of the cost of the post processing that obtains the eigenvectors; this model is based on measurements of the various pieces that make up the postprocessing as these pieces appear elsewhere in the recurrence.

The rate of convergence for the as-yet unconverged eigenvalues is estimated by taking a weighted geometric average of the change in accuracy of the first unconverged Ritz value over the previous five steps. Based on this, we estimate the number of Ritz values that will become accurate enough to be accepted in each of the next four steps. The ratio of the estimated total cost and the estimate of the number of converged eigenvalues is computed for each of the next two steps; we continue if that ratio shows a decrease in average cost. To start the process, we insist that at least ten steps be taken if possible or necessary.

We also use the estimate of the number of eigenvalues converging to decide whether to continue when only a few eigenvalues remain to be computed. If the estimate is that all the eigenvalues needed will be accepted in the next four steps, we continue even if the cost per eigenvalue increases. This is a heuristic to avoid computing a factorization to compute a

very small number of eigenvalues.

The scheme for terminating a run described here was developed over a lengthy period and a large number of experiments. Our experience has been that the cost curve is relatively flat near its minimum, making the choice of where to stop appear to be somewhat flexible. This appearance is misleading; the global minimum we are attempting to reach has been quite sensitive to the local choice. We have chosen to use a scheme that works well for us on a number of examples — we expect it to work well in general, but not to be globally optimal.

To demonstrate the value of a well-tuned dynamic scheme for evaluating cost, we include results of some simple experiments here. We modified our standard scheme to make it terminate early by using 150% of the measured costs instead of the real costs. We maintained the same estimate of the final termination costs, which has the effect of making it appear to be cheaper to stop. We also modified the standard scheme to force it to run five steps beyond where it would normally stop. The results are given in Table 14. These show dramatic sensitivity to small changes in the stopping procedure.

TABLE 7
Comparison of Variations on Termination Model

matrix	Standard Strategy			Termination Early			Termination Late		
	shifts	block steps	cpu cost	shifts	block steps	cpu cost	shifts	block steps	cpu cost
BCSST_26[a]	2	55	152	12	147	242	2	56	181
PLAT1919[b]	48	807	901	68	963	1032	45	824	956

[a] block size 5, lowest 100 modes
[b] block size 3, all modes in [.000025, .24]

4.5. Choice of Blocksize and of Starting Block. The optimal block size for a given problem varies greatly by problem and computer environment. The two largest benefits of the block algorithm are in input/output cost reduction and in treating multiple eigenvalues. However, certain costs, such as obtaining the M-orthogonal factorization and the cost of the eigenanalysis of T_j increase quadratically with the block size p. In general, if multiple eigenvalues are expected and the multiplicities are not too large, it is best to choose a block size as large as the largest expected multiplicity. This is particularly important if many clusters of eigenvalues are expected (Table 11). A block size of six or seven works well in problems with rigid body modes. This is not essential (see Tables 19 and 20) and we rarely find that $p > 10$ is cost-effective.

The effect of input/output cost is considerable. Within the MacNeal-Schwendler NASTRAN product, which runs on a variety of commercial systems, extensive testing resulted in a default block size of seven on all systems. Input and output is particularly expensive within NASTRAN. In an environment in which input/output cost was not measured, a blocksize of 3 was found to be more effective. We provide our results on a small number of experiments in §5; it is likely that these results would change on other systems.

It is always attractive to start an iterative method like the Lanczos algorithm with a good guess at a solution. We begin the first Lanczos run with a randomly generated starting block. Thereafter, we expect to have the approximate eigenvectors (Ritz vectors) from unconverged Ritz values available as estimates of the next eigenvectors to be found. At the time that the eigenvectors of T are available, we do not know where the next shift will be taken. Therefore, we take a starting block built from all of these Ritz vectors. If t vectors are available, each column in the starting block is taken to be the sum of t/p Ritz vectors. We fill the block out randomly when $t < p$. We adopted this approach after extensive experiments comparing various choices of starting blocks, including mixtures of Ritz vectors and random components. We did not find a significant change in the overall cost of the eigensolution in any of the approaches.

5. Experimental Results. The algorithm described in the paper was developed as a general purpose eigensolver for The MacNeal-Schwendler Corporation's structural engineering package NASTRAN [18]. One of the goals in the software design was to make the eigensolver independent of the form of the sparse matrix operations required to represent the matrices involved and their spectral transformations. The key operations needed are matrix-block products, triangular block solves, and sparse factorizations. These, and the data structures representing the matrices, are isolated from the eigensolver. As a result, we have been able to incorporate this code in a number of different environments.

The eigensolver has been used in MSC NASTRAN with two different approaches to the sparse linear equations involved, a profile approach and a sparse multifrontal approach. In both cases the factorization and solve modules are the standard operations of MSC NASTRAN, used directly by the eigensolver. The code has also been incorporated in the PROBE structural engineering package for the Noetics Corporation (now a subsidiary of MSC), the ATLAS structural engineering package developed by Boeing, and the alternative NASTRAN code of Universal Analytics, Inc.. It has also been included in mathematics libraries supplied by Boeing Computer Services (BCSLIB-EXT)[1] and Convex Computer Corporation (Veclib). In all of these implementations the sparse linear equations are solved with vectorized multifrontal codes based on the work in [2, 3, 4]. The multifrontal code does perform numerical pivoting as described in [25].

5.1. The Environment for Experiments. The experiments described in this paper were all performed on a Sun 4/490 workstation with 32 megabytes of main memory. The code is our standard distribution eigensolver; the required matrix factorization, triangular solves and matrix-vector products are the codes in BCSLIB-EXT. The codes are all written in Fortran 77, and were run with the "-O" optimization option of the Sun Fortran compiler. The optimizer is quite effective with the inner loops of the numerical operations. We note that our code is always a block code, even when run with block size 1. This means that our results for block size 1 will be somewhat worse than a single vector code would obtain. The major difference would be in the analysis of the tridiagonal system, where the results of Parlett and Nour-Omid would be available [32], but this would represent only a small savings in general.

[1] BCSLIB-EXT is available at no cost on all Cray Research, Inc. computers.

The test problems are drawn from the symmetric eigenproblems from the Harwell-Boeing Test Collection [11]. These problems were chosen for the collection for use in validating codes, and so often display anomalous or pathological situations. The Harwell-Boeing Test Collection has not kept up with the dramatic increases in problem sizes and computer horsepower, at least in the sense of representing *production* problems. Our code has been used to solve eigenproblems with more than a million degrees of freedom, but the largest problem in the test collection is of order 15,000 and most of the problems are much smaller. As a result, the problem independent costs of the Lanczos algorithm, primarily the analysis of the block tridiagonal systems, are more important than they would be in large production problems.

For most of the examples, we report the costs of the required eigenanalysis as a function of block size; we replicated the analyses for block sizes one to six. For the two largest problems we report the costs in two forms, first in terms of the highest level sparse matrix primitives, and then as a breakdown in terms of the functional operations of the code. For the latter we report statistics as outlined in Table 8.

TABLE 8
Cost Statistics

task	elements
recurrence	solve $(K - \sigma M)U_j = MQ_j$ compute $U_j = U_j - Q_{j-1}B_j^T$ compute $R = U_j - Q_jA_j$
factorization	LDL^T factorization of $K - \sigma M$
reorthogonalization	M-orthogonalization of R local reorthogonalization external selective orthogonalization partial reorthogonalization
block tridiagonal	eigenanalysis of block tridiagonal matrix at each block step
eigenvector	eigenanalysis of block tridiagonal matrix at final block step and back transformation with Lanczos vectors
start up	generating M-orthogonal starting block, including projection into correct subspace and initial external selective orthogonalization
shift selection	choose new shift

5.2. Some Empirical Examples. Throughout this paper we have used some of the problems from the Harwell/Boeing test collection [11] to demonstrate particular aspects of our algorithms. We close by using a slightly larger subset to illustrate some of the global behavior of the code, particularly as it concerns aspects over which the user exercises control. We chose six test problems, all but one of which were collected from actual industrial or research applications. Because ordinary problems are less likely to have been called to

our attention, these problems are probably unusual. Nonetheless they provide useful demonstrations.

TABLE 9
Test Problems

matrix	order	nonzeros in K	nonzeros in M	description
BCSST_08	1074	7017	1074	television station
BCSST_24	3562	81736	3562	Calgary Saddledome
BCSST_25	15439	133840	15439	76 story skyscraper
BCSST_26	1992	16129	1922	nuclear reactor containment floor
BCSST_34	588	11003	12429	MSC Nastran buckling problem
PLAT1919	1919	17159	—[a]	Atlantic and Indian Oceans

[a] ordinary eigenvalue problem

Two of the problems have been used as the primary examples in this paper. They are BCSST_26, a model of a nuclear reactor containment floor used for seismic analysis, and PLAT1919, a finite difference model of tidal currents. These models both were included in the test collection because of the large number of eigenpairs that were required of each. In both cases the number of modes is large because the analysis depended on knowing *all* of the modes in specified intervals. At the time of their accession to the collection these matrices represented very large and difficult problems. Increases in computer speeds and the use of appropriate algorithms make these problems appropriate for contemporary workstations.

Apart from the number of eigenpairs required, the nuclear reactor containment floor problem appears to have no unusual characteristics. Details of the eigenanalysis, as a function of blocksize, are given in Table 10. These results exhibit a pattern that is common to all of the problems: The number of factorizations and Lanczos runs decreases rapidly as the block size increases; the cost of the eigenanalysis initially decreases as well, but then increases. This reflects the fact that as the blocksize increases, the length of the Lanczos runs increase in terms of the dimension of $\mathcal{Q}_j$. Longer runs involve greater costs, particularly for maintaining semi-orthogonality and for the back transformation of the eigenvectors. For these relatively small matrices, the costs of longer runs begins to dominate the costs of factoring and applying the matrix operators rather early. For reference a analysis using only a single Lanczos run with a block size of three had a cost of 671 for this problem, nearly three times the cost of the analysis with shifting.

The oceanography problem has several additional characteristics that made it sufficiently interesting to warrant special attention. First, the eigenvalues desired are very much in the interior of the spectrum. There are 818 eigenvalues above and 465 eigenvalues below the values we want. This problem was analyzed, without use of the spectral transformation, in [5, 22]. Without shifting, it was barely possible to compute the eigenvalues in the interval [.0001, .24]; the eigenvalues in [.000025, .0001] were also of interest, but were impossible to compute. Secondly, all the eigenvalues, except a singleton at zero, are positive and occur in pairs. These multiple eigenvalues play havoc with an ordinary, point, Lanczos algorithm.

TABLE 10
Computation of 200 Eigenvalues from BCSST_26 (Shift Statistics)

block size	cpu cost	number of factor-izations	runs	block steps	block solves
1	385	44	75	954	1177
2	233	24	23	357	425
3	253	20	19	261	317
4	321	14	13	197	235
5	420	12	12	183	218
6	348	4	4	87	98

With either a block size of 1 or 2, it is difficult for a code to be sure that it has exhibited the full multiplicities of the eigenvalues — the shifting strategy must be prepared to assist. Even with shifting, the single vector code of Ericsson and Ruhe [12, 15] was unable to cope with the multiplicities of the eigenvalues [24].

Table 11 demonstrates a number of effects of the choice of blocksize. It is clearly important to choose a blocksize large enough to cope with the multiplicities expected. Blocksizes of 1 or 2 result in a large number of reruns to pick up second copies of eigenvalues. This is shown by the discrepancy between the number of factorizations and the number of runs. Although the reruns incur no new cost for factorizations, the need for a rerun is detected only when the second end of a subinterval has been used as the shift; this will probably not be the best shift for the values near the other end of the subinterval. Each of these reruns also requires more extensive use of external selective orthogonalization than would an ordinary run. The increase in the number of runs and of external selective orthogonalization generates unusually high start up costs for these low blocksizes. In addition, the large number of eigenpairs requested is reflected in unusual costs for eigenvector computation, which also reflects the heavy input/output cost of this step.

TABLE 11
Computation of 636 Eigenvalues from PLAT1919 (Shift Statistics)

block size	cpu cost	number of factor-izations	runs	block steps	block solves
1	1395	95	209	2831	3248
2	1360	85	152	1871	2168
3	915	48	48	807	901
4	968	29	29	558	615
5	1187	17	18	393	428
6	1282	12	12	301	324

Three of the problems are rather ordinary. BCSST_08 is a model of a building housing television studios and other production and administrative facilities. Its claim to fame is

the presence of isolated double and near triple eigenvalues. At one time it was considered to have very close eigenvalues. The lowest 24 eigenvalues are given in Table 12. The close eigenvalues cause relatively slow convergence, which causes our code to make a number of short runs. Details are given in Table 13. This problem can be solved easily enough without our elaborate shifting strategy; it would appear that our code may be shifting unnecessarily. In fact, shifting is appropriate. We removed the cost termination from the recurrence, so that the code would terminate only when it believes it has computed all the requested eigenvalues. The cost of doing so was 65.0 seconds for a unit blocksize run and 99.4 seconds for a block size of three. Shifting can be effective.

TABLE 12
Lowest 26 Eigenvalues of BCSST_08

i	λ_i	i	λ_i	i	λ_i
1	6.900	9	91.05	17	138.7
2	18.14206	10	93.45	18	139.6
3	18.1423664462086	11	130.9	19	140.6
4	18.1423664462086	12	131.5	20	141.1
5	84.78615	13	132.9	21	141.566
6	84.7864335537914	14	136.2	22	141.638
7	84.7864335537914	15	137.2	23	142.19
8	85.54	16	138.4	24	142.642

TABLE 13
Computation of lowest 20 Eigenvalues from BCSST_08 (Shift Statistics)

block size	cpu cost	number of factor-izations	runs	block steps	block solves
1	41.8	11	11	123	151
2	35.4	6	6	66	83
3	31.3	4	4	46	57
4	31.1	4	3	36	44
5	38.0	4	3	34	42
6	47.0	4	3	34	42

BCSST_24 is a structural model of the Olympia Hockey Arena in Calgary, Alberta, Canada. The main interest in this model is the fact the eigenanalysis of this particular model confirmed the need for a redesign of the structure prior to construction. Table 14 gives summary statistics, while Table 15 gives the detailed breakdown of cost.

The only unusual aspect of these analyses is that the eigenvalues of this problem, although not particularly close in general, are close enough that more than a single run is used for smaller blocksizes. The 26 lowest eigenvalues are given in Table 16; there is one near-pair of eigenvalues, but the rest are singletons. The poor performance for blocksize one is probably due to an artifact of the program that is not tuned to the unit blocksize case. The code

TABLE 14
Computation of lowest 20 Eigenvalues from BCSST_24 (Shift Statistics)

block size	cpu cost	number of			
		factor-izations	runs	block	
				steps	solves
1	351	10	16	179	226
2	125	3	3	42	50
3	123	2	2	33	38
4	139	3	2	24	29
5	148	2	2	24	29
6	143	2	1	17	19

TABLE 15
Computation of lowest 20 Eigenvalues from BCSST_24 (Cost Breakdown)

Percent of Cost of Eigenextraction							
block size	recur-rence	factor-ization	re-orthog.	block tridiag.	eigen-vector	start up	shift select.
1	27	53	6	0	1	11	1
2	37	44	8	0	3	6	1
3	45	30	14	1	5	4	1
4	38	40	11	1	4	5	1
5	45	25	15	1	7	5	0
6	41	26	18	2	9	3	1

TABLE 16
Lowest 26 eigenvalues of BCSST_24

i	λ_i
1	4.33
2	9.47
3	11.8
4	14.1
5	17.5
6	20.4
7	20.8
8	23.8
9	27.6

i	λ_i
10	29.7
11	36.1
12	37.5
13	37.6
14	39.0
15	40.8
16	41.6
17	47.1
18	51.9

i	λ_i
19	54.8
20	58.0
21	60.9
22	63.0
23	63.1
24	66.17
25	66.24
26	76.0

is forced to take a minimum of ten steps in each run that does not break down, but the code will stop if no eigenvalue has converged in those ten steps nor appears likely to do so in the next two steps. The standard Lanczos recurrence does not obtain rapid enough convergence on this problem to prevent this from occurring, but probably would work adequately if more steps were taken.

BCSST_34 is a buckling problem obtained from The MacNeal Schwendler Corporation, as an example with eigenvalues of both signs. (The other buckling problems in the collection happen to be of structures that had not yet buckled — all the eigenvalues are positive.) Table 17 describes the shifts taken for three different specifications of modes. The lowest positive eigenvalue is 5.595 $\times 10^3$; the smallest negative mode is -2.012 $\times 10^4$.

TABLE 17
Shift History for BCSST_34, block size 3

problem	shift	number of steps	number of eigenvalues
lowest 10 modes	-8.78 $\times 10^1$	19	11
	-3.16 $\times 10^4$	10	0
	2.93 $\times 10^4$	5	2
lowest mode	-8.78 $\times 10^1$	10	0
	7.34 $\times 10^3$	7	1
	-2.36 $\times 10^4$	9	1
lowest negative mode	-8.78 $\times 10^1$	10	0
	-2.36 $\times 10^4$	7	1

BCSST_25 is a seismic model of the Columbia Center, a 76 story skyscraper in Seattle, Washington. This model was collected in 1981, at which time it was too large to be run in-core on any computer available to the authors. It is a sign of the progress in computing that ten years later this problem can be solved on engineering workstations. We continue to retain this problem because it is pathologically difficult. The lowest 132 eigenvalues are listed in Table 18. For reference, the largest eigenvalue of this structure is 1.51 $\times$ 10^8.

The smallest eigenvalues are nearly negligible when compared to the largest eigenvalue and they are very close to one another. Our code determines clusters of eigenvalues based on its accuracy tolerance, which defaults to 2.31 $\times$ 10^{-11} in IEEE arithmetic. We apply this tolerance to the transformed eigenvalues, which are *not* close enough to be treated as a cluster or even as two clusters and four isolated values. (Note that if we applied the tolerance to the untransformed eigenvalues, all of these values would be a cluster, which is not appropriate.) As a result this problem counters our usual shifting strategy — in this case we must take a shift very close to the eigenvalues in order to overcome the very poor separation and slow convergence. This situation, eigenvalues almost, but not quite in a cluster, represents a worst case. When run in default mode (10 lowest eigenvalues, no information on location), our code is able to resolve the four lowest modes, but our attempts to avoid shifting directly on an eigenvalue prevent it from shifting close enough to resolve the next two "clusters" before it terminates on a fail-safe mechanism. (Five consecutive runs are unable to resolve any of the missing "six" eigenvalues, although they do compute some of the higher modes.)

TABLE 18
Lowest 132 Eigenvalues of BCSST_25

i	λ_i
1	9.6140×10^{-4}
2	9.7948×10^{-4}
3	9.7961×10^{-4}
4	9.8380×10^{-4}
5	9.85801×10^{-4}
⋮	⋮
68	9.85803×10^{-4}
69	9.86240×10^{-4}
⋮	⋮
132	9.86243×10^{-4}

These runs do provide enough inertia information to locate the lowest 132 modes in the interval $[9.5 \times 10^{-4}, 9.9 \times 10^{-4}]$. The experiments in the Tables 19 and 20 are follow-on runs using that location information. The failure with block size 1 demonstrates the extra power of the block algorithm. (This problem can be run with block size 1 by choosing interval specifications yet closer to the eigenvalues.)

TABLE 19
Computation of 132 Eigenvalues from BCSST_25 (Shift Statistics)

block size	cpu cost	number of factor-izations	runs	block steps	block solves
1	failed	5	5	69	82
2	6381	17	17	295	344
3	3417	5	5	116	129
4	3725	5	5	100	113
5	4115	4	4	81	92
6	8339	8	8	134	158

5.3. Summary. The results in the previous section illustrate some of the characteristics of the shifted block Lanczos algorithm. Only problems BCSST_24 and BCSST_25 are large enough to begin to demonstrate the behavior of the algorithm on large problems. We can make some general statements about the change in the behavior as problem sizes increase. We expect to see the cost of the factorization and linear equation solutions to increase faster than linearly. Assuming that the eigenvalue distributions do not change, the cost of reorthogonalization, of generating the starting block and of the eigenvector computation will increase linearly with the change in problem size. The block tridiagonal eigenanalysis and the shift selection should remain constant and their contributions to cost will become even

TABLE 20
Computation of lowest 132 Eigenvalues from BCSST_25 (Cost Breakdown)

Percent of Cost of Eigenextraction							
block size	recur- rence	factor- ization	re- orthog.	block tridiag.	eigen- vector	start up	shift select.
2	27	46	13	0	4	9	0
3	31	25	27	1	13	2	0
4	32	23	28	1	13	2	0
5	30	17	37	1	13	2	0
6	30	17	36	0	6	12	0

smaller. We note that the cost of the necessary reorthogonalizations is an important fraction of the cost — this is a strong argument for preserving only *semi-orthogonality* rather than complete orthogonality. We remind the reader that our cost measures include a factor for input/output traffic, an essential ingredient in preserving semi-orthogonality.

The reader will see the difficulty in making an *a priori* choice of blocksize. The advantages of the block algorithm are clearly demonstrated, but there is no optimal choice for block size. A choice of three is always good on these problems on our Sun workstation, but is likely to be less than optimal for a vibration problem with six rigid body modes. Systems that impose higher costs for input/output will make higher blocksizes more effective. So will larger problem sizes, because the additional costs imposed by larger blocksizes are in terms that are independent of the problem size.

These issues should be kept in the perspective of the power of the spectral transformation. None of the problems described here is solvable in any practical sense using the naive reduction to standard form. For example, the oceanography problem, PLAT1919, was analyzed in [5, 22] without any transformation — what were really the desired eigenvalues were not close to appearing after N steps. (In unreported experiments, $3N$ steps had resulted in little improvement.) Although it is possible to solve some of the simpler problems by inverting the problem, as in (2), this is clearly not sufficient for all of the problems. The oceanography problem, PLAT1919, is singular, so some nontrivial shift is required. Even with a shift at the lower endpoint, .000025, a single Lanczos run to compute the 200 eigenvalues above this point had a cost of 6637 (for blocksize 5). In contrast, our standard shifted code had a cost of 1187 for computing all 636 desired eigenvalues with the same blocksize. The Columbia Center model has the same characteristics. The naive reduction would result in a problem with separations of 10^{-13} for the "well-separated" eigenvalues; even the simple reciprocal transformation would be clearly inadequate to begin to solve this problem. It is only with the combined power of the block Lanczos algorithm and the spectral transformation that we can solve these problems in a reasonable time.

6. Acknowledgments. The authors would like to thank David Scott for his participation in the initial design of this code, and Thomas Ericsson, Bahram Nour-Omid and Beresford Parlett for many enlightening discussions during the preparation of this code. We also want to thank Louis Komszik and The MacNeal-Schwendler Corporation for their

support of this work.

REFERENCES

[1] *The Boeing Extended Mathematical Subprogram Library*, Boeing Computer Services, P.O. Box 24346, Seattle, WA 98124-0346, 1989.

[2] C. C. ASHCRAFT, *A vector implementation of the multifrontal method for large sparse symmetric positive linear systems*, Tech. Rep. ETA-TR-51, Boeing Computer Services, 1987.

[3] C. C. ASHCRAFT AND R. G. GRIMES, *The influence of relaxed supernode partitions on the multifrontal method*, ACM Trans. on Math. Software, 15 (1989), pp. 291–309.

[4] C. C. ASHCRAFT, R. G. GRIMES, J. G. LEWIS, B. W. PEYTON, AND H. D. SIMON, *Progress in sparse matrix methods for large linear systems on vector supercomputers*, Internat. J. Supercomput. Appl., 1 (1987), pp. 10–30.

[5] A. CLINE, G. GOLUB, AND G. PLATZMAN, *Calculations of normal modes of oceans using a Lanczos method*, in Sparse Matrix Computations, J. Bunch and D. Rose, eds., Academic Press, 1976, pp. 409 – 426.

[6] J. CULLUM AND W. E. DONATH, *A block Lanczos algorithm for computing the q algebraically largest eigenvalues and a corresponding eigenspace of large sparse real symmetric matrices*, in Proc. of the 1974 IEEE Conference on Decision and Control, Phoenix, 1974.

[7] J. CULLUM AND R. WILLOUGHBY, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. 1 Theory*, Birkhäuser, Boston, 1985.

[8] ——, *Lanczos Algorithms for Large Symmetric Eigenvalue Computations, Vol. 2 Users Guide*, Birkhäuser, Boston, 1985.

[9] ——, *Computing eigenvalues of large matrices, some Lanczos algorithms and a shift and invert strategy*, in Advances in Numerical Partial Differential Equations and Optimization: Proc. of 5th Mexico-United States Workshop, S. Gomez, J.P. Hennart, and R.A. Tapia, eds., Proc. in Applied Mathematics, Vol. 47, SIAM, Philadelphia 1991.

[10] J. W. DANIEL, W. B. GRAGG, L. KAUFMAN, AND G. W. STEWART, *Reorthogonalizations and stable algorithms for updating the Gram-Schmidt QR factorization*, Math. Comp., 30 (1976), pp. 772 – 795.

[11] I. S. DUFF, R. G. GRIMES, AND J. G. LEWIS, *Sparse matrix test problems*, ACM TOMS, 15 (1989), pp. 1 – 14.

[12] T. ERICSSON, *User Guide for STLM*, Tech. Rep. UMINF-96.82, University of Umea, 1982.

[13] ——, *A generalized eigenvalue problem and the Lanczos algorithm*, in Large Scale Eigenvalue Problems, J. Cullum and R. Willoughby, eds., North-Holland, 1986.

[14] T. ERICSSON AND A. RUHE, *The spectral transformation Lanczos method*, Math. Comp., 34 (1980), pp. 1251 – 1268.

[15] ——, *STLM - a Software Package for the Spectral Transformation Lanczos Algorithm*, Tech. Rep. UMINF-101.82, University of Umea, Sweden, 1982.

[16] B. S. GARBOW, J. M. BOYLE, J. J. DONGARRA, AND C. B. MOLER, *Matrix Eigensystem Routines - EISPACK Guide Extension*, vol. 51 of Lecture Notes in Computer Sciences, Springer-Verlag, Berlin, 1977.

[17] G. GOLUB AND R. UNDERWOOD, *The block Lanczos method for computing eigenvalues*, in Mathematical Software III, J. Rice, ed., Academic Press, New York, 1977.

[18] R. GRIMES, J. LEWIS, L. KOMZSIK, D. SCOTT, AND H. SIMON, *Shifted block Lanczos algorithm in MSC/NASTRAN*, in Proceedings of the MSC/NASTRAN User's Conference, Los Angeles, 1985.

[19] R. GRIMES, J. LEWIS, AND H. SIMON, *Eigenvalue Problems and algorithms in Structural Engineering*, in Large Scale Eigenvalue Problems, J. Cullum and R. Willoughby, eds., North-Holland, 1986.

[20] M. T. JONES AND M. L. PATRICK, *LANZ: Software Solving the Large Sparse Symmetric Generalized Eigenproblem*, Tech. Rep. NAS1-18605, ICASE, NASA Langley Res. Center, Hampton, VA, August 1990.

[21] C. LANCZOS, *An Iteration Method for the Solution of Eigenvalue Problem of Linear Differential and Integral Operators*, J. Res. Nat. Bur. Stand., 45 (1950), p. 255.

[22] J. LEWIS, *Algorithms for Sparse Matrix Eigenvalue Problems*, PhD thesis, Stanford University, Dept. of Computer Science, 1977.
[23] J. LEWIS AND R. GRIMES, *Practical Lanczos algorithms for solving structural engineering eigenvalue problems*, in Sparse Matrices and their Uses, I. Duff, ed., Academic Press, London, 1981.
[24] J. LEWIS AND H. SIMON, *Numerical Experience with the Spectral Transformation Lanczos Method*, Tech. Rep. ETA-TR-16, Boeing Computer Services, 1983.
[25] J. W.-H. LIU, *A partial pivoting strategy for sparse symmetric matrix decomposition*, ACM Trans. on Math. Software, 13 (1987). 173-182.
[26] M. NEWMAN AND P. FLANAGAN, *Eigenvalue Extraction in NASTRAN by the Tridiagonal Reduction (FEER) Method — Real Eigenvalue Analysis*, NASA Contrator Rep. CR-2731, NASA, 1976.
[27] B. NOUR-OMID, *The Lanczos Algorithm for Solution of Large Generalized Eigenproblems*, in The Finite Element Method, T. Hughes, ed., Prentice-Hall, Englewood Cliffs, New Jersey, 1987.
[28] B. NOUR-OMID, B. N. PARLETT, T. ERICSSON, AND P. S. JENSEN, *How to implement the spectral transformation*, Math. Comp., 48 (1987), pp. 663 - 673.
[29] C. C. PAIGE, *The Computation of Eigenvalues and Eigenvectors of Very Large Sparse Matrices*, PhD thesis, University of London, 1971.
[30] ——, *Error analysis of the Lanczos algorithm for tridiagonalizing a symmetric matrix*, J. Inst. Math. Appl., 18 (1976), pp. 341 - 349.
[31] B. PARLETT, *The Symmetric Eigenvalue Problem*, Prentice Hall, Englewood Cliffs, New Jersey, 1980.
[32] B. N. PARLETT AND B. NOUR-OMID, *The use of refined error bounds when updating eigenvalues of a growing symmetric tridiagonal matrix*, Linear Algebra and its Applications, 68 (1985), pp. 179 - 219.
[33] B. PARLETT, B. NOUR-OMID, AND Z. A. LIU, *How to Maintain Semi-orthogonality Among Lanczos Vectors*, Tech. Rep. PAM-420, Center for Pure and Applied Math., University of California, Berkeley, CA, 1988
[34] B. PARLETT AND D. SCOTT, *The Lanczos algorithm with selective orthogonalization*, Math. Comp., 33 (1979), pp. 217 - 238.
[35] Y. SAAD, *On the rates of convergence of the Lanczos and block-Lanczos methods*, SIAM J. Num. Anal., 17 (1980), pp. 687 - 706.
[36] D. SCOTT, *Block Lanczos Software for Symmetric Eigenvalue Problems*, Tech. Rep. ORNL/CSD 48, Oak Ridge Nat. Lab., 1979.
[37] ——, *The advantages of inverted operators in Rayleigh-Ritz approximations*, SIAM J. Sci. Stat. Computing, 3 (1982), pp. 68 - 75.
[38] H. SIMON, *Analysis of the symmetric Lanczos algorithm with reorthogonalization methods*, Lin. Alg. Appl., 61 (1984), pp. 101 - 131.
[39] ——, *The Lanczos algorithm with partial reorthogonalization*, Mathematics of Computation, 42 (1984), pp. 115 - 136.
[40] B. T. SMITH, J. M. BOYLE, J. J. DONGARRA, B. S. GARBOW, Y. IKEBE, V. C. KLEMA, AND C. B. MOLER, *Matrix Eigensystem Routines - EISPACK Guide*, vol. 6 of Lecture Notes in Computer Sciences, Springer-Verlag, Berlin, 1976.

A. Matrix Inertias. We need to interpret the number of negative eigenvalues of $K - \sigma M$ and $K - \sigma K_\delta$ in terms of the eigenvalues of the original vibration or buckling problems. The result we want to prove is:

Interpretation of $\nu(K - \sigma M)$ or $\nu(K - \sigma K_\delta)$

vibration analysis:	
M positive definite	# of eigenvalues $< \sigma$
M positive semidefinite	(# of eigenvalues $< \sigma$) $+ \gamma$
	$\gamma = \begin{cases} 0 & \text{some cases} \\ \dim(\mathcal{N}(M)) & \text{other cases} \end{cases}$
buckling analysis:	
K positive definite	# of eigenvalues in $(0, \sigma)$ or $(\sigma, 0)$
K positive semidefinite	(# of eigenvalues in $(0, \sigma)$ or $(\sigma, 0)$) $+ \gamma$
	$\gamma = \begin{cases} 0 & \sigma \text{ of one sign} \\ \dim(\mathcal{N}(K)) & \sigma \text{ of other sign} \end{cases}$

From this we conclude that:

$$\nu(K - \sigma_2 M) - \nu(K - \sigma_1 M) = \text{number of eigenvalues in } (\sigma_1, \sigma_2)$$

where we assume that $\sigma_2 > \sigma_1$. In the case of buckling analyses we further assume that both σ_1 and σ_2 have the same sign.

There are four cases, which will be considered in pairs. In all cases we assume that the problem is a well-posed generalized symmetric eigenproblem, i.e., that there exists some linear combination $\alpha K + \beta M$ that is positive definite.

A.1. $Kx = \lambda Mx$ with M positive definite. We can apply the obvious reduction to standard form. The eigenvalues of $Kx = \lambda Mx$ are the same as the eigenvalues of $C = L_M^{-1} K L_M^{-T}$, where L_M is the Cholesky factor of M. It follows that the number of eigenvalues of C less than σ is the same as the number of eigenvalues of $Kx = \lambda Mx$ less than σ. But $C - \sigma I$ is congruent to $L_M(C - \sigma I)L_M^T$ and this is simply $K - \sigma M$. Thus, the number of negative terms in the decomposition of $K - \sigma M$ is the number of eigenvalues less than σ. Obviously, the interpretation of the inertia has the same meaning here as in the ordinary eigenvalue problem.

A.2. $Kx = \lambda Mx$ with M positive semidefinite. The problem in which M is singular is more complex, in that signs must be assigned to the infinite eigenvalues. Suppose that M is positive semidefinite, with p zero eigenvalues $\omega_1, \omega_2, \ldots, \omega_p = 0$, and $n - p$ positive eigenvalues $\omega_{p+1}, \ldots, \omega_n$. M is a symmetric matrix, so it has an eigendecomposition

$$M = S \Omega S^T.$$

Let the nonsingular matrix W be defined by

$$W = \Psi S^T,$$

where Ψ is the diagonal matrix with $\psi_{ii} = 1$ for $i = 1, 2, \ldots, p$ and $\psi_{ii} = 1/\sqrt{\omega_i}$ for $i = p + 1, \ldots, n$. Then the eigenvalues of

$$KX = MX\Lambda$$

are the same as the eigenvalues of

$$WKW^TY = WMW^TY\Lambda,$$

where $Y = W^{-T}X$. By definition of W,

$$WMW^T = \Psi S^T S\Omega S^T S\Psi,$$

which, because S is orthogonal, reduces to

$$WMW^T = \Psi\Omega\Psi.$$

By the construction of Ψ, $\Psi\Omega\Psi$, and hence WMW^T, is the two by two block partitioned matrix

$$\begin{pmatrix} 0 & 0 \\ 0 & I \end{pmatrix},$$

where I is an $(n-p) \times (n-p)$ identity matrix. Partition $WKW^T = C$ conformally as

$$\begin{pmatrix} C_{11} & C_{21}^T \\ C_{21} & C_{22} \end{pmatrix}.$$

Let y be an eigenvector of $(WKW^T)y = \lambda(WMW^T)y$, partitioned conformally as $[y_1, y_2]^T$. Then y is an eigenvector if and only if

$$C_{11}y_1 + C_{21}^T y_2 = 0 \tag{33}$$

and

$$C_{21}y_1 + C_{22}y_2 = \lambda y_2. \tag{34}$$

Under the assumption that some linear combination $\alpha K + \beta M$ is positive definite, it follows that $\alpha(WKW^T) + \beta(WMW^T)$ is positive definite. But a positive definite matrix has positive definite principal minors, and the (1,1) block of the transformed linear combination is αC_{11} (α is clearly not zero because M is not definite). Therefore, C_{11} is a definite matrix, certainly nonsingular. Equation (33) then implies that

$$y_1 = -C_{11}^{-1}C_{21}^T y_2.$$

Substituting in (34), we obtain

$$(C_{22} - C_{21}C_{11}^{-1}C_{21}^T)y_2 = \lambda y_2.$$

Thus, the finite eigenvalues of $Kx = \lambda Mx$ are the eigenvalues of the Schur complement of C_{11} in C.

By Sylvester's theorem the inertia of $(K - \sigma M)$ is the same as the inertia of $WKW^T - \sigma WMW^T$. But the partitioned form of the LDL^T decomposition of $WKW^T - \sigma WMW^T$

has as its (1,1) block the decomposition of C_{11}, and as its (2,2) block the decomposition of $(C_{22} - C_{21}C_{11}^{-1}C_{21}^T) - \sigma I$. $(C_{22} - C_{21}C_{11}^{-1}C_{21}^T) - \sigma I$ is the valuable part of the decomposition, but the inertia for the entire matrix is offset by the inertia of the (1,1) block. The offset is constant – it describe the sign given to the infinite eigenvalues. That all of the infinite eigenvalues have the same sign is due to the fact that a positive definite linear combination of K and M exists, that is, that the problem is a generalized *symmetric* eigenproblem[13]. The difference between $\nu(K - \sigma_1 M)$ and $\nu(K - \sigma_2 M)$ will still be the number of eigenvalues in $[\sigma_1, \sigma_2)$, since the constant term cancels.

Further, in vibration analysis, we know that both K and M are positive semidefinite. It follows that both α and β will be positive when M is only semidefinite. The positive semidefiniteness of K then implies that C_{11} is a positive definite matrix, so $\nu(C_{11}) = 0$. Thus, the inertia of the factored matrix retains the exact same meaning for the positive semidefinite vibration case as for the positive definite case.

A.3. $Kx = \lambda K_\delta x$ with K positive definite. In buckling analysis, we do not have any definiteness properties for K_δ but we do for K. The assumption that K is positive definite allows us to invert the problem. Thus

$$Kx = \lambda K_\delta x$$

implies

$$K_\delta x = (1/\lambda) K x = \mu K x,$$

and all the eigenvalues μ in the second equation are finite. This transformed problem is in the standard (K_δ, K) form in which the right hand side matrix, K, is positive definite. We will apply our previous analysis to determine the number of eigenvalues of (K_δ, K) that lie in the image of the interval of interest in the original problem. Thus, to determine the number of eigenvalues of $Kx = \lambda K_\delta x$ less than σ, we will instead determine the number of eigenvalues of the inverted problem (K_δ, K) in the interval(s) in the variable $\mu = 1/\lambda$ that corresponds to the interval $(-\infty, \sigma)$ in the variable λ.

There are three subcases that must be considered. The first is the case $\sigma = 0$. But the interval $(-\infty, 0)$ for λ is mapped to the interval $(-\infty, 0)$ in $1/\lambda$. Thus, the number of negative eigenvalues of $Kx = \lambda K_\delta x$ is the same as the number of eigenvalues of (K_δ, K) less than 0. By the standard result above, this is simply the number of negative eigenvalues of $K_\delta, \nu(K_\delta)$.

The second case is the case $\sigma < 0$. The transformation from λ to $1/\lambda$ transforms σ to $1/\sigma$. The number of eigenvalues in $(-\infty, \sigma)$ is the same as the number of eigenvalues of (K_δ, K) in the interval $(1/\sigma, 0)$. By our previous results this is

$$\nu(K_\delta) - \nu(K_\delta - (1/\sigma)K),$$

which, because σ is negative, is the same as

$$\nu(K_\delta) - \nu(K - \sigma K_\delta).$$

Note that the number of eigenvalues between σ and 0 is simply $\nu(K - \sigma K_\delta)$.

The third case is $\sigma > 0$. In this case, the interval $(-\infty, \sigma)$ in λ must be treated as the union of the interval $(-\infty, 0)$ and the interval $[0, \sigma)$. There are $\nu(K_\delta)$ eigenvalues in the first subinterval. The second subinterval is transformed into $(1/\sigma, +\infty)$. This is

$$\nu(K_\delta) + \pi(K_\delta - (1/\sigma)K)$$

or

$$\nu(K_\delta) + \nu(K - \sigma K_\delta).$$

It follows even in this case that $\nu(K - \sigma K_\delta)$ is the number of eigenvalues between 0 and σ.

The buckling problem will have infinite eigenvalues if K_δ is singular. However, the signs of these eigenvalues are irrelevant to the interpretation of the inertias because the interpretation always considers only finite subintervals.

A.4. $Kx = \lambda K_\delta x$ with K positive semidefinite. The most general case we consider is a buckling analysis in which K is only positive semidefinite. For this case to be a symmetric generalized problem there must exist some pair (α, β) with $\alpha K + \beta K_\delta$ positive definite. By the positive semidefiniteness of K, we may assume that α is 1. (We can add any positive multiple of K without changing the positive definiteness of the sum.) We will combine the analysis for the M semidefinite case together with the positive definite buckling case above to obtain similar results. The difficulty in this case is that we need to assign signs to the zero eigenvalues.

We assign signs to the zero eigenvalues by redoing the analysis of §A.2, applying the same reductions to K instead of M. K is a symmetric matrix, so it has an eigendecomposition

$$K = \hat{S}\hat{\Omega}\hat{S}^T,$$

with p zero eigenvalues $\widehat{\omega_1}, \widehat{\omega_2}, \ldots, \widehat{\omega_p} = 0$, and $n - p$ positive eigenvalues $\widehat{\omega_{p+1}}, \ldots, \widehat{\omega_n}$. As before, let the nonsingular matrix $\widehat{W}$ be defined by

$$\widehat{W} = \hat{\Psi}\hat{S}^T,$$

where $\hat{\Psi}$ is the diagonal matrix with $\widehat{\psi_{ii}} = 1$ for $i = 1, 2, \ldots, p$ and $\widehat{\psi_{ii}} = 1/\sqrt{\widehat{\omega_i}}$ for $i = p + 1, \ldots, n$. Then the eigenvalues of

$$KX = K_\delta X \Lambda$$

are the same as the eigenvalues of

$$\widehat{W}K\widehat{W}^T Y = \widehat{W}K_\delta\widehat{W}^T Y\Lambda,$$

where $Y = \widehat{W}^{-T}X$. Again

$$\widehat{W}K\widehat{W}^T = \hat{\Psi}\hat{S}^T\hat{S}\hat{\Omega}\hat{S}^T\hat{S}\hat{\Psi},$$

reduces to

$$\widehat{W}K\widehat{W}^T = \hat{\Psi}\hat{\Omega}\hat{\Psi}^{\cdot} = \begin{pmatrix} 0 & 0 \\ 0 & I \end{pmatrix}.$$

Partition $\widehat{W}K_\delta\widehat{W}^T = E$ conformally as

$$\begin{pmatrix} E_{11} & E_{21}^T \\ E_{21} & E_{22} \end{pmatrix}.$$

Let y be an eigenvector of $(\widehat{W}K\widehat{W}^T)y = \lambda(\widehat{W}K_\delta\widehat{W}^T)y$, partitioned conformally as $[y_1, y_2]^T$. Then y is an eigenvector if and only if

$$E_{11}y_1 + E_{21}^T y_2 = 0 \tag{35}$$

and

$$\lambda(E_{21}y_1 + E_{22})y_2 = y_2.$$

As before the (1,1) block of the transformed linear combination, βE_{11}, is a positive definite matrix. Equation (35) then implies that

$$y_1 = -E_{11}^{-1}E_{21}^T y_2,$$

or

$$\lambda(E_{22} - E_{21}E_{11}^{-1}E_{21}^T)y_2 = y_2.$$

Thus, the finite, nonzero, eigenvalues of $Kx = \lambda K_\delta x$ are the reciprocals of the nonzero eigenvalues of the Schur complement of E_{11} in E.

The partitioned form of the LDL^T decomposition of $\widehat{W}K\widehat{W}^T - \sigma\widehat{W}K_\delta\widehat{W}^T$ has as its (1,1) block the decomposition of $-\sigma E_{11}$, and as its (2,2) block the decomposition of $I - \sigma(E_{22} - E_{21}E_{11}^{-1}E_{21}^T)$. The Schur complement block is in the form of §A.3, since the identity matrix is clearly positive definite. Again, the inertia of the full matrix is the inertia of $I - \sigma(E_{22} - E_{21}E_{11}^{-1}E_{21}^T)$ offset by the inertia of the (1,1) block. Notice that the offset depends on the sign of the shift – it describes the signs of the eigenvalues of $-\sigma E_{11}$. Because E_{11} is definite, either all the eigenvalues of $-\sigma E_{11}$ are positive or all are negative. Thus, the offset will be zero for shifts of one sign and nonzero for shifts of the other sign. Still, the difference between $\nu(K - \sigma_1 K_\delta)$ and $\nu(K - \sigma_2 K_\delta)$ will still be the number of eigenvalues in $[\sigma_1, \sigma_2)$, as long as both shifts have the same sign.

Unfortunately $\nu(E_{11})$ does interfere with the interpretation of the number of eigenvalues between σ and 0 for shifts on one side of the origin. However, $\nu(E_{11})$ is the dimension of the nullspace of K, a quantity that is often known adventitiously. If not, it can be estimated by factoring $K - \rho I$, where ρ is chosen smaller than the least nonzero eigenvalue of K, but large enough so that the factorization is stable.